

VB.NET & Access Database Connectivity

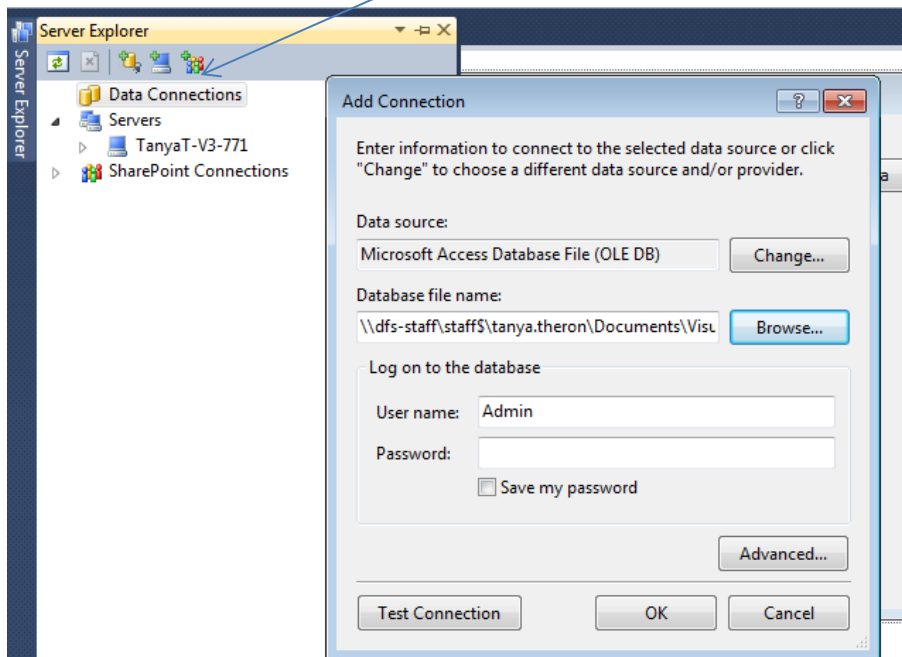
Table of Contents

CONNECTING TO FROM VB.NET FORM TO ACCESS.....	2
Connect to database (only needs to be done once for your project).....	2
Create a string variable for the connection to the database (for each form).....	2
Opening and closing the console	4
MANIPULATING DATA IN THE DATABASE FROM VB.NET FORMS	5
Adding Records	5
Changing Existing Records	7
Deleting Records	7
Retrieving Existing Records.....	7
All records in a table	7
Specific records in a table	7
Datasets	8
To process records in a dataset row by row	8
To reference individual row and elements	8
SHOWING SELECTED DATA ON FORMS	9
Using DataGridView	9
Using ReportViewer	10
USEFUL CODE SNIPPETS.....	11
Using the Mouse to select data from a DataGridView	11
SQL SELECT to find a maximum value.....	12
SQL SELECT for rows containing a specific value	13

CONNECTING TO FROM VB.NET FORM TO ACCESS

Connect to database (only needs to be done once for your project)

Right click on 'Data Connections' or click

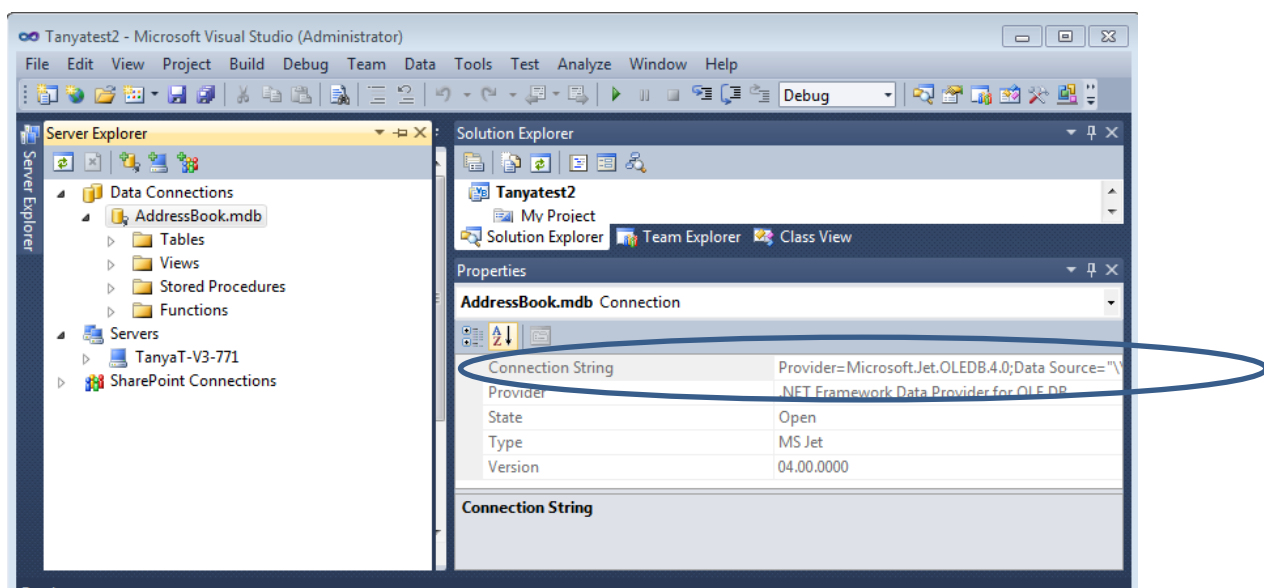


Select the datasource to be Microsoft Access Database File (OLE DB) then browse to the database file (good practice to keep in same folder as vb project)

Test connection.

Create a string variable for the connection to the database (for each form)

STEP 1 Click on the database. In the properties window you will see a 'connection string'. Copy the connection string (ctrl+c).



STEP 2 Double click on your form. If your form is blank this code will appear:

```
Form1 (Declarations)
Public Class Form1
    Private Sub Form2_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    End Sub
End Class
```

STEP 3

You need to type **Imports System.Data.OleDb** above the top line. (When you type the . a list of methods will appear) This will import the Ole Database module (like importing random in Python)

```
Form1 (Declarations)
Imports System.Data.OleDb
Public Class Form1
    Private Sub Form2_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    End Sub
End Class
```

STEP 4

Now you need to establish a connection to the database. Make sure it is saved in the same directory as your project.

Declare a new string variable called **connstring**. Assign the value to be the value in the connection string property of the data source (from step 1). However remove the path (as the db is saved in the same directory as your project). For example if the connection string is **Provider=Microsoft.Jet.OLEDB.4.0;DataSource=\\dfsstaff\staff\$\tanya.theron\Documents\Visual Studio 2010\Projects\Tanyatest2\AddressBook.mdb** you will remove the bold directory details so it will become **"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=AddressBook.mdb"**.

This would be your line of code.

```
Public connstring as String = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=AddressBook.mdb"
```

Now you need to declare a variable for a connection object

```
Public conn as new OleDbConnection(connstring)
```

This is how your code will look (new lines circled):

```
Imports System.Data.OleDb

Public Class Form2
    Public connstring As String = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=AddressBook.mdb"
    Public conn As New OleDbConnection(connstring)

    Private Sub Form2_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    End Sub
End Class
```

Opening and closing the console

Anytime you will want to open the console to accept SQL commands. This will be within button routines.

```
conn.Open()
```

After you have finished executing SQL statements, **close the console**.

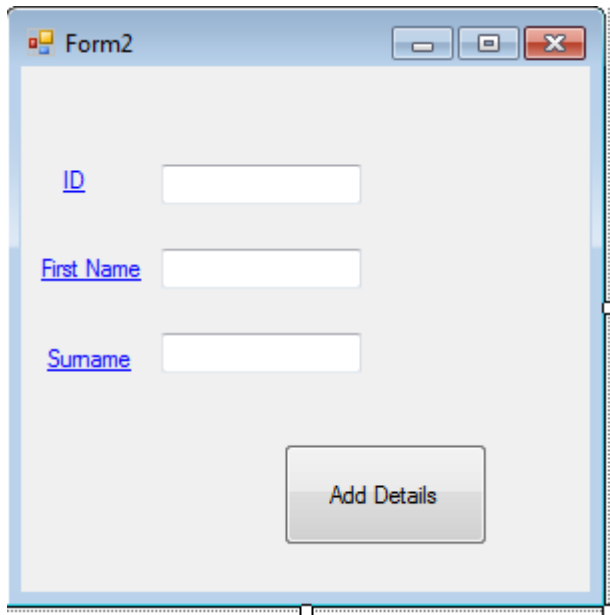
```
conn.Close() (otherwise it wastes system resources).
```

MANIPULATING DATA IN THE DATABASE FROM VB.NET FORMS

Adding Records

In this simple example I will build on the above example to show you how to add data from text boxes into your database. First I will create three text boxes on the form and a button to execute the code. I will call the text boxes txtID, txtFirstName and txtLastName

Here is the form:



Double clicking on the button takes me to the sub-routine for the button:

```
Button1 Click
Imports System.Data.OleDb

Public Class Form1

    Public connstring As String = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=AddressBook.mdb"
    Public conn As New OleDbConnection(connstring)

    Private Sub Form2_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        End Sub
End Class
```

Code for button goes here!

When the button is clicked we want to add the text from the text boxes to the fields in the database. Here is how:

- 1) `conn.open()` – opens the connection
- 2) `Dim SqlQuery As String = "INSERT INTO tblContacts (ID,FirstName,Surname) VALUES (@ID,@FirstName,@Surname)"` 'declares sql statement as string variable to be used further down the code.
- 3) `Dim SqlCommand As New OleDbCommand` 'declares OleDbCommand variable

4) With SqlCommand

```
.CommandText = SqlQuery  
.Parameters.AddWithValue("@ID", TxtID.Text)  
.Parameters.AddWithValue("@FirstName", TxtFirstName.Text)  
.Parameters.AddWithValue("@Surname", TxtSurname.Text)  
.Connection = conn  
.ExecuteNonQuery()
```

End With

conn.Close() *Close the connection*

Highlighted yellow is always the same.

Highlighted green is the sql statement – note the values with the @ sign are parameters from the text boxes – these are declared as part of the SqlCommand, highlighted in turquoise.

Highlighted turquoise are the parameters (values being taken from your text boxes or form objects that are used in the sql statement)

Here's the code:

```
Imports System.Data.OleDb  
  
Public Class Form2  
    Public connstring As String = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=AddressBook.mdb"  
    Public conn As New OleDbConnection(connstring)  
  
    Private Sub Form2_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load  
        End Sub  
  
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click  
        conn.Open()  
        Dim SqlQuery As String = "INSERT INTO tblContacts (ID,FirstName,Surname) VALUES (@ID,@FirstName,@Surname)"  
        Dim SqlCommand As New OleDbCommand  
        With SqlCommand  
            .CommandText = SqlQuery  
            .Parameters.AddWithValue("@ID", txtID.Text)  
            .Parameters.AddWithValue("@FirstName", TxtFirstName.Text)  
            .Parameters.AddWithValue("@Surname", txtLastName.Text)  
            .Connection = conn  
            .ExecuteNonQuery()  
        End With  
        conn.Close()  
    End Sub  
End Class
```

Changing Existing Records

Same process for deleting or updating a record, just change the sql statement and parameters:

```
Dim SqlQuery As String = "UPDATE tblStock SET Quantity = @Quantityvariable WHERE StockID = @StockIDvariable"

Dim SqlCommand As New OleDbCommand
With SqlCommand
    .CommandText = SqlQuery
    .Parameters.AddWithValue("@Quantityvariable", TextBox5.Text)
    .Parameters.AddWithValue("@StockIDvariable", TextBox4.Text)
    .Connection = conn
    .ExecuteNonQuery()
End With
conn.Close()
End Sub
```

Deleting Records

```
Dim SqlQuery As String = "DELETE FROM tblStock WHERE StockID=@StockIDvariable"

Dim SqlCommand As New OleDbCommand
With SqlCommand
    .CommandText = SqlQuery
    .Parameters.Add(New OleDbParameter("@StockIDvariable", TextBox3.Text))
    .Connection = conn
    .ExecuteNonQuery()
End With
conn.Close()
End Sub
```

Retrieving Existing Records

All records in a table

In a SEL statement * is a wildcard meaning all. So this example selects all fields and records from a table and populated a dataset table

```
Dim SqlQuery As String = "SELECT * FROM tblStock "

'create data adapter
Dim da As OleDbDataAdapter = New OleDbDataAdapter(SqlQuery, conn)

'create dataset
Dim ds As DataSet = New DataSet

'fill dataset
da.Fill(ds, "Stock")

'get data table
Dim dt As DataTable = ds.Tables("Stock")
```

Specific records in a table

SQL will not be able to recognise VB variables, we need to pass variables in line. This example passes a value from a TextBox into the SELECT statement.

```
'Taking parameter from TextBox6
Dim SqlQuery As String = "SELECT * FROM tblStock WHERE StockID = '" & TextBox6.Text & "' "
```

Datasets

In Python you used arrays to hold database records in memory. In Visual Basic .NET it is more common to use a structure called a Dataset. Think of this as a temporary database table in memory. To **create a new dataset** in code,

```
Dim SqlQuery As String = "SELECT * FROM tblStock "

'create data adapter
Dim da As OleDbDataAdapter = New OleDbDataAdapter(SqlQuery, conn)

'create dataset
Dim ds As DataSet = New DataSet


'fill dataset
da.Fill(ds, "Stock")

'get data table
Dim dt As DataTable = ds.Tables("Stock")
```

To process records in a dataset row by row. This For loop will execute it's code for every row in the dataset.

```
'get data table
Dim dt As DataTable = ds.Tables("Stock")

'process individual rows - this is an example
For Each row As DataRow In dt.Rows
    StockTotal = StockTotal + row.Item(1)
Next
```



Item(1) would be the second field (or column) in the dataset Table. They start with 0 being the first field.

To reference individual row and elements in a dataset. You can explicitly reference rows in a dataset table as well as the field. Both start with 0 as the first row and field.

```
'first row and first element
TextBox1.Text = ds.Tables("Stock").Rows(0).Item(0)
```


SHOWING SELECTED DATA ON FORMS

Using DataGridView

To **execute a SELECT command and fill a DataGridView** from a dataset. A DataGridView is a handy way of showing records on a form.

It is selected from the Toolbox under Data and dragged onto the form.

	Seat Name	Performance	Price
▶	A1	F	7.5
	A2	F	7.5
	A3	F	7.5
	A4	F	7.5
	A5	F	7.5
	A6	F	7.5
	B1	S	7.5
	B2	S	7.5
	B3	S	7.5
*			

The circled code will fill the datagridview.

```
Private Sub Button7_Click(sender As Object, e As EventArgs) Handles Button7.Click
    conn.ConnectionString = connstring
    If conn.State = ConnectionState.Closed Then
        conn.Open()
    End If

    'Taking parameter from TextBox6
    Dim SqlQuery As String = "SELECT * FROM tblStock WHERE StockID = '" & TextBox6.Text & "'"

    Dim da As OleDbDataAdapter = New OleDbDataAdapter(SqlQuery, conn)
    Dim ds As DataSet = New DataSet
    da.Fill(ds, "Stock")
    Dim dt As DataTable = ds.Tables("Stock")
    conn.Close()

    'Fill 2nd DataGrid with Dataset table following SQL statement above
    With DataGridView2
        .AutoGenerateColumns = True
        .DataSource = ds
        .DataMember = "Stock"
    End With
End Sub
```

Using ReportViewer

To fill Reportviewer with an entire table from a database. Report viewer is a more graphical way to present output of records. It can be customised and has a print functionality.

```
Private Sub FormForReport_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    'TODO: This line of code loads data into the 'TEST_DBDataSet1.tblStock' table
    Me.tblStockTableAdapter.Fill(Me.TEST_DBDataSet1.tblStock)
    Me.ReportViewer1.RefreshReport()
End Sub
```

To execute a SELECT command and fill Reportviewer from a dataset,

Firstly, there is no datatable at the time of design – only when the code runs. So, we must create a placeholder dataset for the report.

1. click : project --> add new item -->(on data template) choose DataSet.
2. right click on empty space --> Add --> DataTable
3. right click on DataTable --> Add --> Column, (add the columns you need to match your dataset, remember the correct order and datatypes).
4. click : project --> add new item -->(on reporting template) choose ReportWizard.
5. on DataSet combobox, choose DataSet1 that we create before, click next.
6. hold click and drag your columns from step 3 from the left side to the Values field, next.
7. choose your header style then finish.
8. go to Form2, select your ReportViewer, on right top corner, click little arrow there, and choose Report1.rdlc

Secondly, fill the report with the dataset. e.g.

```
Private Sub Button8_Click(sender As Object, e As EventArgs) Handles Button8.Click
    conn.ConnectionString = connstring
    If conn.State = ConnectionState.Closed Then
        conn.Open()
    End If

    'Taking parameter from TextBox6
    Dim SqlQuery As String = "SELECT * FROM tblStock WHERE StockID = '" & TextBox6.Text & "'"

    Dim da As OleDbDataAdapter = New OleDbDataAdapter(SqlQuery, conn)
    Dim ds As DataSet = New DataSet
    da.Fill(ds, "Stock")
    Dim dt As DataTable = ds.Tables("Stock")
    conn.Close()

    Me.ReportViewer1.LocalReport.DataSources.Item(0).Value = dt
    Me.ReportViewer1.RefreshReport()
End Sub
```

USEFUL CODE SNIPPETS

Using the Mouse to select data from a DataGridView

This code snippet allows you to **select an item of data from a datagridview**. This example uses a datagridview called DataGridView1 but your code may be different. Also, the selected content is put into a TextBox just for this example

```
Private Sub DataGridView1_CellContentClick(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles DataGridView1.CellContentClick

    Dim value As Object = DataGridView1.Rows(e.RowIndex).Cells(e.ColumnIndex).Value

    If IsDBNull(value) Then
        TextBox4.Text = "" ' blank if dbnull values
    Else
        'CType to convert the values to Strings for the TextBox
        TextBox4.Text = CType(value, String)
    End If
End Sub
```

This alternative code snippet allows you to select an item of data from a datagridview **and reference individual fields in the selected row**. Again, this example uses a datagridview called DataGridView1 but your code may be different. The selected content is put into a set of Textbox's just for this example

```
Private Sub DataGridView1_CellContentClick(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles DataGridView1.CellContentClick
    Dim value As Object = DataGridView1.Rows(e.RowIndex).Cells(e.ColumnIndex).Value
    If IsDBNull(value) Then
        TextBox4.Text = "" 'blank if dbnull values
    Else
        'CType to convert the values to Strings for the TextBox
        'eRowIndex is selected datagrid row
        '0,1,2 refer to the field column index
        TextBox4.Text = CType(DataGridView1(0, e.RowIndex).Value, String)
        TextBox5.Text = CType(DataGridView1(1, e.RowIndex).Value, String)
        TextBox6.Text = CType(DataGridView1(2, e.RowIndex).Value, String)
    End If
End Sub
```

SQL SELECT to find a maximum value

This code snippet allows you to select a row **based on the maximum value in a field**. This example uses a nested SELECT statement.

Note the SqlQuery string. The main SELECT statement is looking for the row with the highest value in the TestNumber field. It achieves this with another SQL SELECT statement.

The nested SELECT is using the MAX command to only select the row with the highest value.

```
conn.Open()
'find latest test
Dim SqlQuery As String = "SELECT * FROM tblTests WHERE TestNumber = (SELECT MAX(TestNumber) FROM tblTests)"
Dim da As OleDbDataAdapter = New OleDbDataAdapter(SqlQuery, conn)
Dim ds As DataSet = New DataSet
da.Fill(ds, "Tests")
Dim dt As DataTable = ds.Tables("Tests")

Dim Score As Integer = 0
If Ans1.Text = ds.Tables("Tests").Rows(0).Item(11) Then
    Score = Score + 2
End If
If Ans2.Text = ds.Tables("Tests").Rows(0).Item(12) Then
    Score = Score + 2
End If
If Ans3.Text = ds.Tables("Tests").Rows(0).Item(13) Then
    Score = Score + 2
End If
```

SQL SELECT for rows containing a specific value

Rather than selecting all data in a table (SELECT * FROM tblUsers), this snippet selects records based on the criteria of one value. In this case the value in a TextBox is matched against the LoginID field in the database table.

Note the exact syntax as the different speech marks and spacing is important. It must be in this format as SQL statements do not easily recognise the variables we use within our code.

```
conn.Open()

'Fill the DataSet with users
Dim SqlQuery As String = "SELECT * FROM tblUsers WHERE LoginID = '" & TextBox1.Text & "'"
Dim da As OleDbDataAdapter = New OleDbDataAdapter(SqlQuery, conn)
Dim SqlCommand As New OleDbCommand
Dim ds As DataSet = New DataSet
da.Fill(ds, "Users")
Dim dt As DataTable = ds.Tables("Users")

'Display correct menu
If ds.Tables("Users").Rows(0).Item(4) = TextBox2.Text Then
    If ds.Tables("Users").Rows(0).Item(3) = "T" Then
        frmTeacherMenu.Show()
    ElseIf ds.Tables("Users").Rows(0).Item(3) = "S" Then
        frmStudentMenu.Show()
    End If
Else
    MsgBox("Incorrect Password")
End If

conn.Close()
```

Note how the individual fields in the selected row are referenced.

Rows(0) is the selected row we are looking at

Items(3) is the 4th field in the selected row (remember they start at zero from the left)