THE LECTURE 15

FUNDAMENTALS OF DATABASE CONNECTIVITY

DATABASE CONNECTIVITY

C# and .Net can work with a majority of databases, the most common being Oracle and Microsoft SQL Server. But with every database, the logic behind working with all of them is mostly the same.

- Connection – To work with the data in a database, the first obvious step is the connection. The connection to a database normally consists of the below-mentioned parameters.
- Database name or Data Source – The first important parameter is the database name to which the connection needs to be established. Each connection can only work with one database at a time.
- Credentials – The next important aspect is the username and password which needs to be used to establish a connection to the database. It ensures that the username and password have the necessary privileges to connect to the database.
- Optional parameters - For each database type, you can specify optional parameters to provide more information on how .net should handle the connection to the database. For example, one can specify a parameter for how long the connection should stay active. If no operation is performed for a specific period of time, then the parameter would determine if the connection has to be closed.

DATABASE CONNECTIVITY

Selecting data from the database – Once the connection has been established, the next important aspect is to fetch the data from the database. C# can execute 'SQL' select command against the database. The 'SQL' statement can be used to fetch data from a specific table in the database.

Inserting data into the database – C# can also be used to insert records into the database. Values can be specified in C# for each row that needs to be inserted into the database.

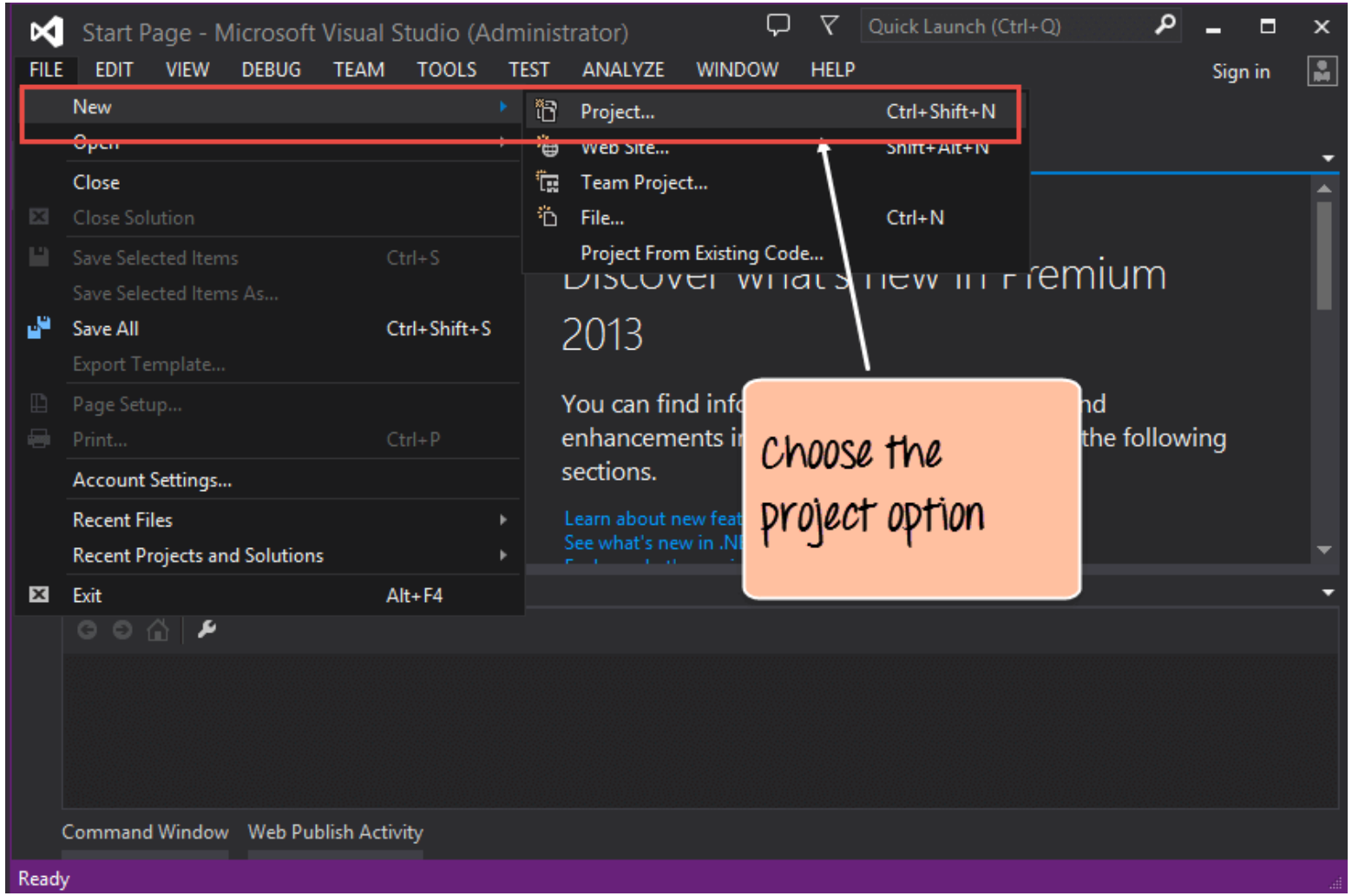
Updating data into the database – C# can also be used to update existing records into the database. New values can be specified in C# for each row that needs to be updated into the database.

Deleting data from a database – C# can also be used to delete records into the database. Select commands to specify which rows need to be deleted can be specified in C#.

HOW TO CONNECT C# TO DATABASE

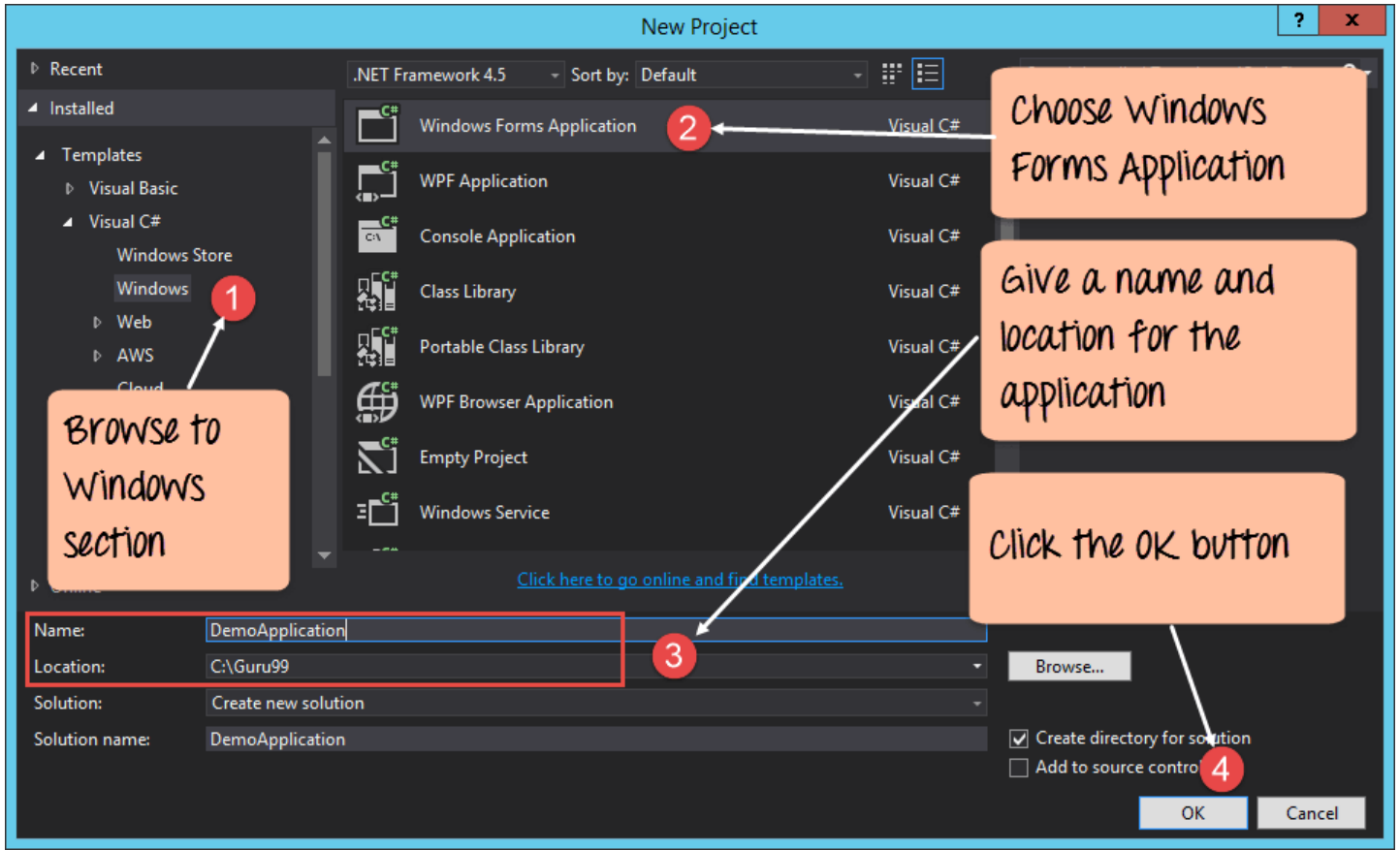
- Let's now look at the code, which needs to be kept in place to create a connection to a database. In our example, we will connect to a database which has the name of Demodb. The credentials used to connect to the database are given below
- **Username – sa**
- **Password – demo123**
- We will see a simple Windows forms application to work with databases. We will have a simple button called "Connect" which will be used to connect to the database.
- So let's follow the below steps to achieve this
- **Step 1)** The first step involves the creation of a new project in Visual Studio. After launching Visual Studio, you need to choose the menu option New->Project.

CREATING A PROJECT



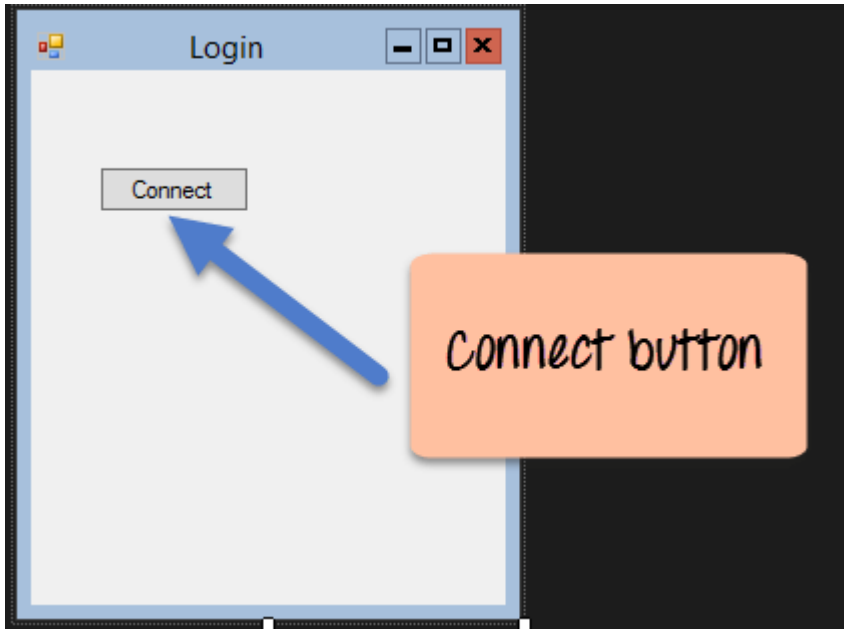
CREATING A PROJECT

- **Step 2)** The next step is to choose the project type as a Windows Forms application. Here, we also need to mention the name and location of our project.



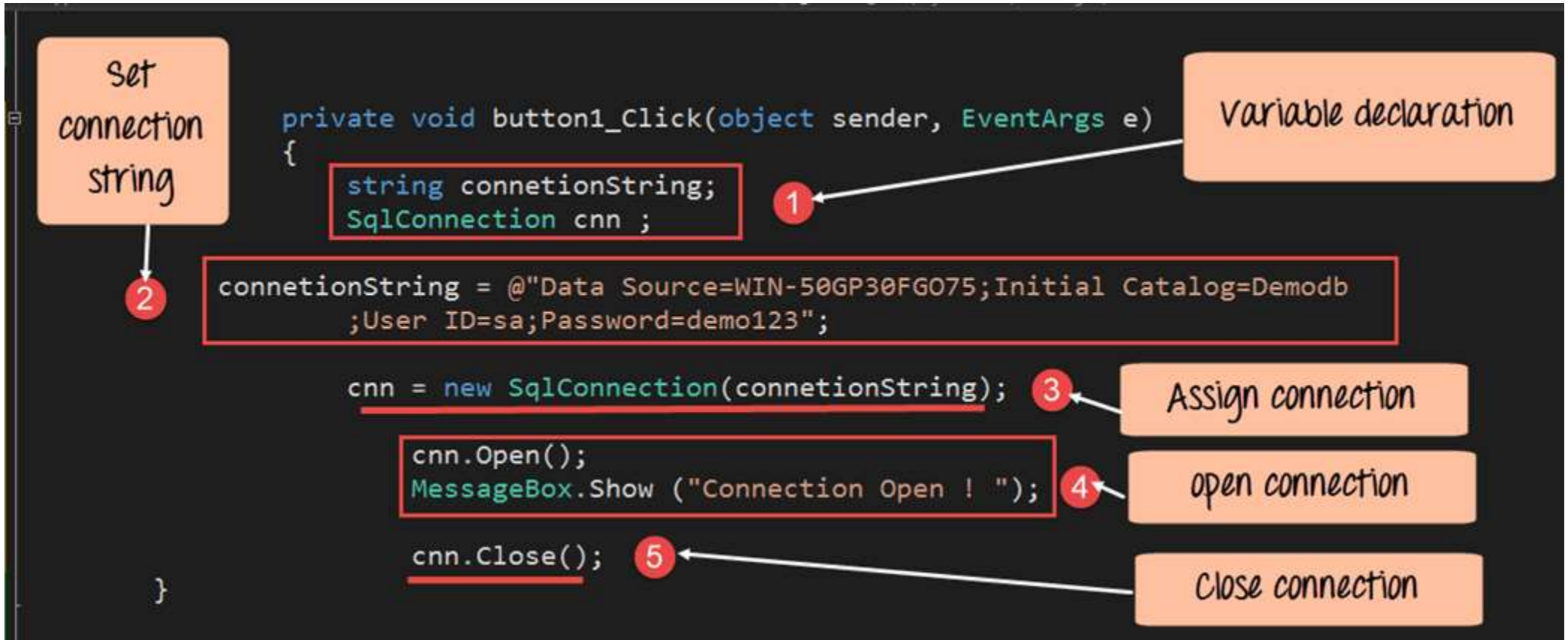
CONNECTION ELEMENT

- **Step 3)** Now add a button from the toolbox to the Windows form. Put the text property of the Button as Connect. This is how it will look like



CONNECTION STRING

- Now double click the form so that an event handler is added to the code for the button click event. In the event handler, add the below code

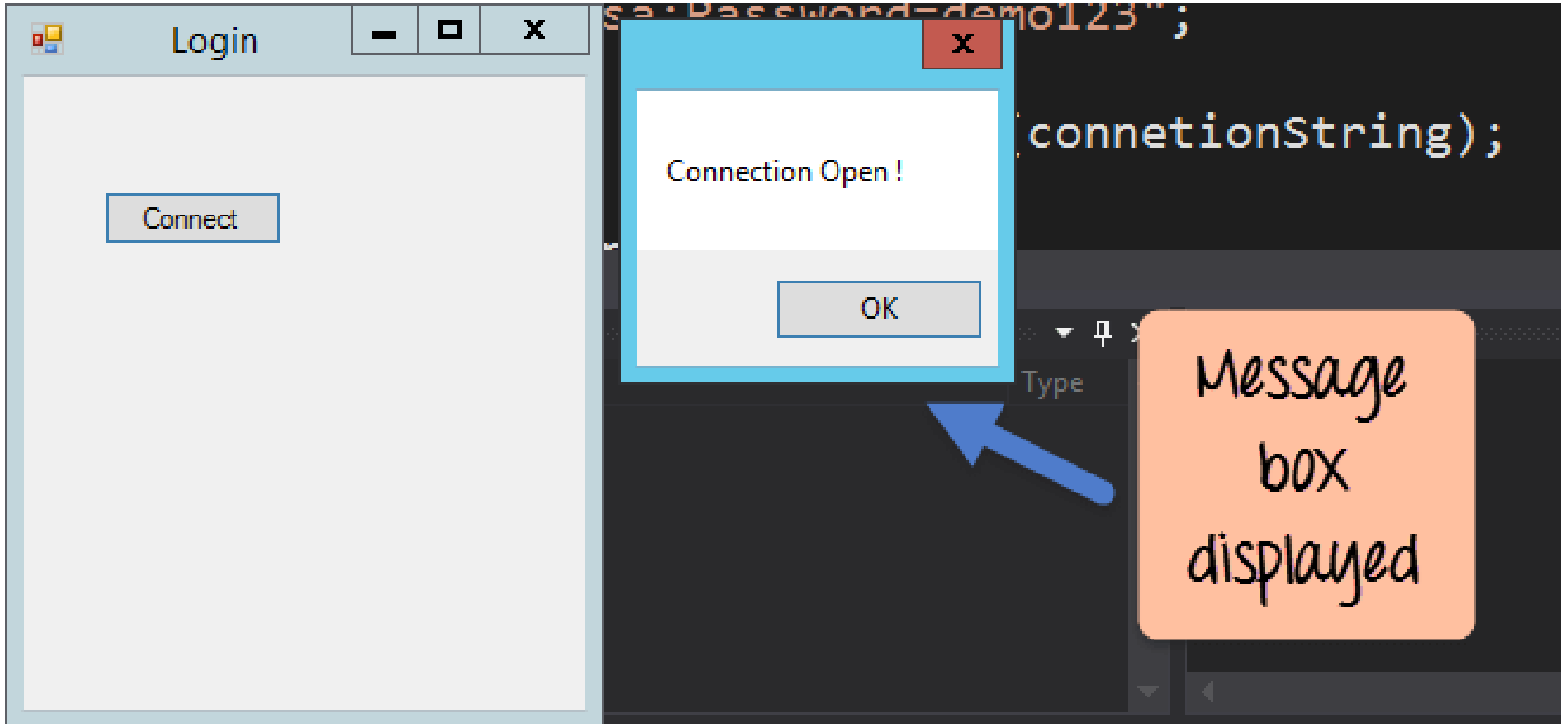


CONNECTION STRING

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace DemoApplication1
{
 public partial class Form1 : Form
 {
  public Form1()
  {
   InitializeComponent();
  }

  private void button1_Click(object sender, EventArgs e)
  {
   string connetionString;
   SqlConnection cnn;
   connetionString = @"Data Source=WIN-50GP30FGO75;Initial Catalog=Demodb;User ID=sa;Password=demol23";
   cnn = new SqlConnection(connetionString);
   cnn.Open();
   MessageBox.Show("Connection Open  !");
   cnn.Close();
  }}}
```

CONNECTION WINDOW



ACCESS DATA WITH THE SQLDATAREADER

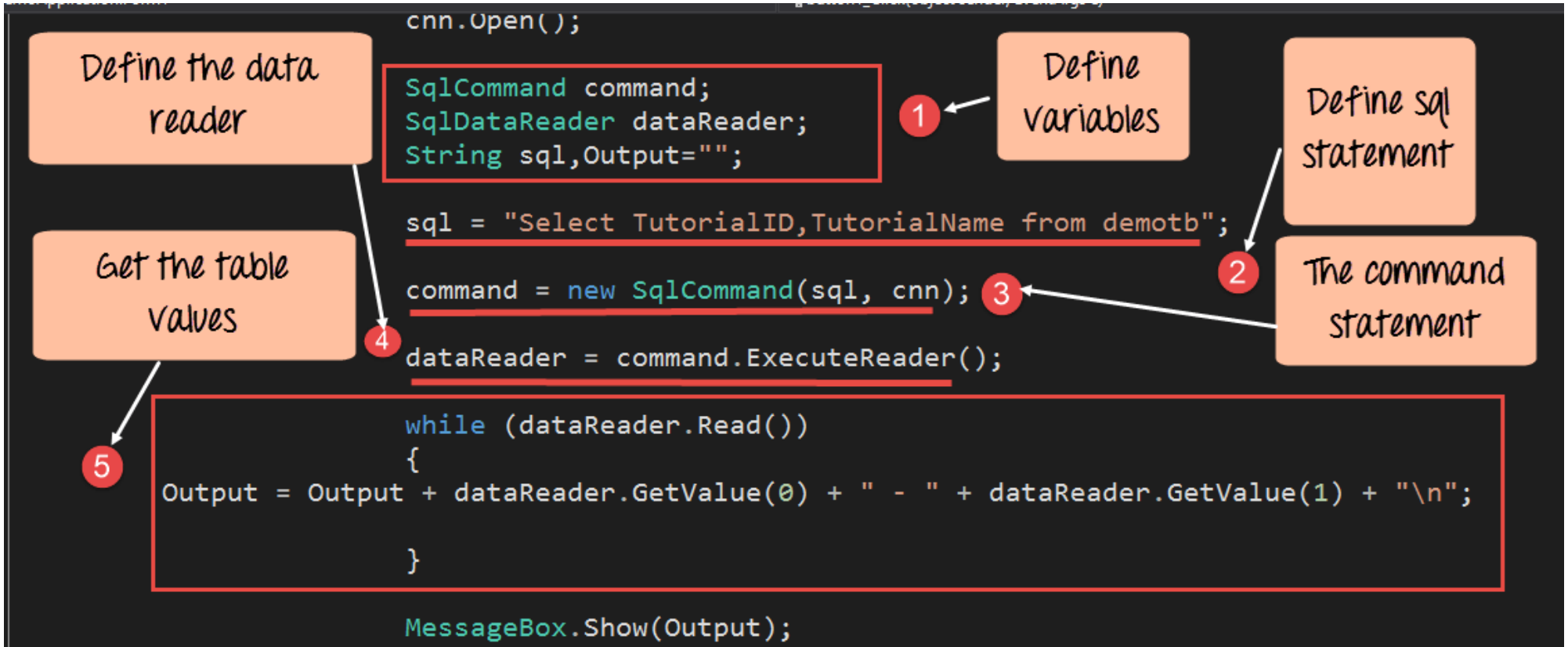
- To showcase how data can be accessed using C#, let us assume that we have the following artifacts in our database.

1. A table called demotb. This table will be used to store the ID and names of various Tutorials.
2. The table will have 2 columns, one called "TutorialID" and the other called "TutorialName."
3. For the moment, the table will have 2 rows as shown below.

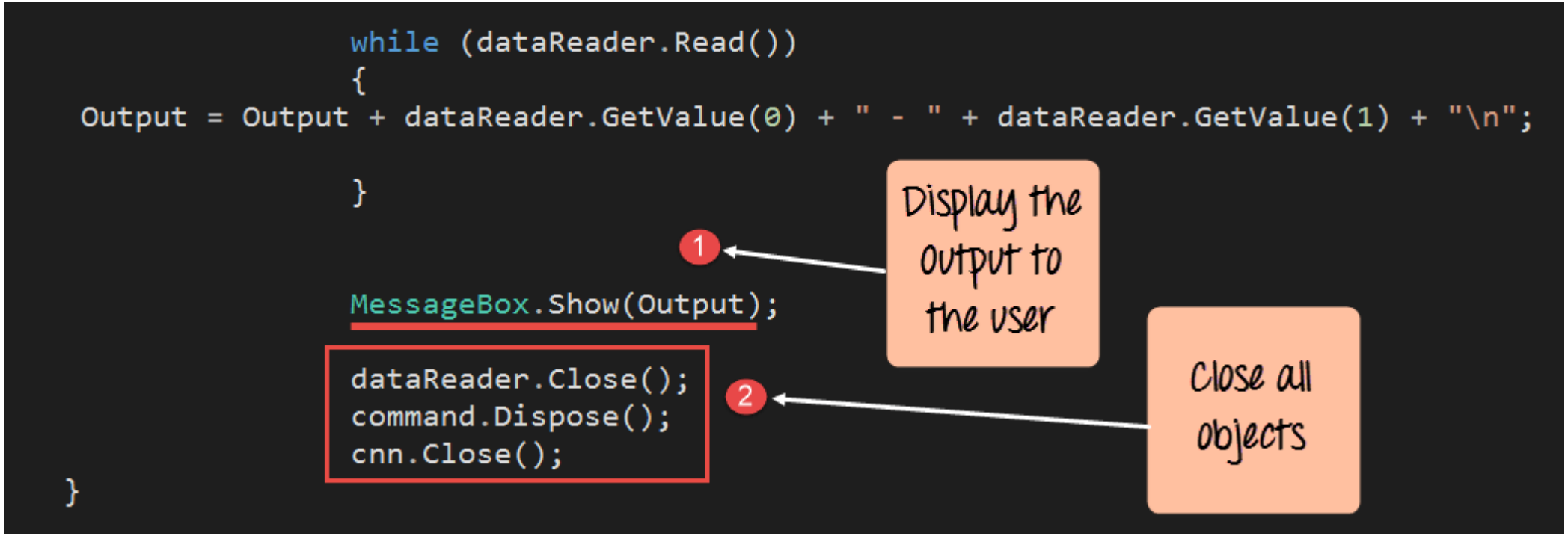
TutorialID	TutorialName
1	C#
2	ASP.Net

ACCESS DATA WITH THE SQLDATAREADER

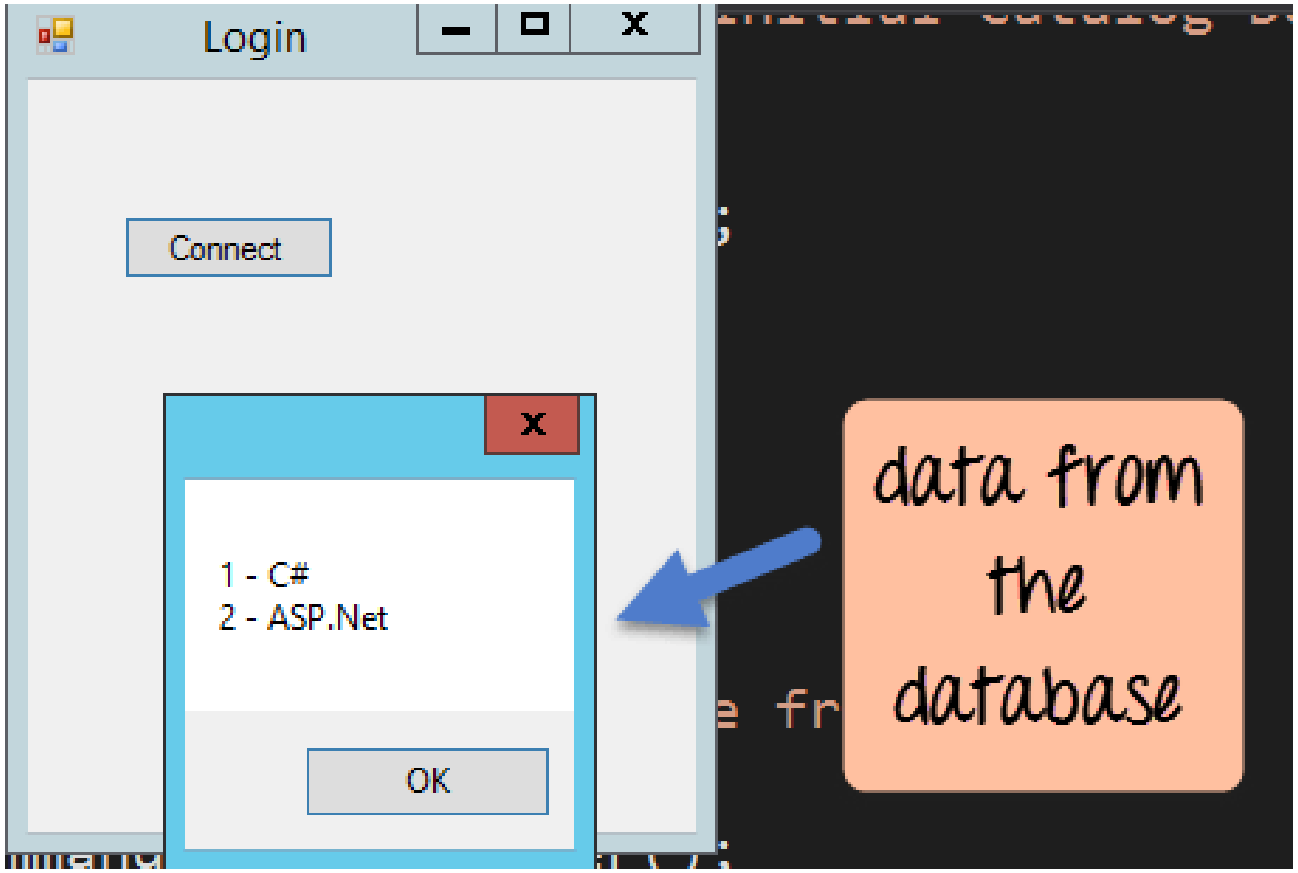
- Let's split the code into 2 parts so that it will be easy to understand for the user.
- The first will be to construct our "select" statement, which will be used to read the data from the database.
- We will then execute the "select" statement against the database and fetch all the table rows accordingly.



- In the final step, we will just display the output to the user and close all the objects related to the database operation.



OUTPUT WINDOW



C# INSERT INTO DATABASE

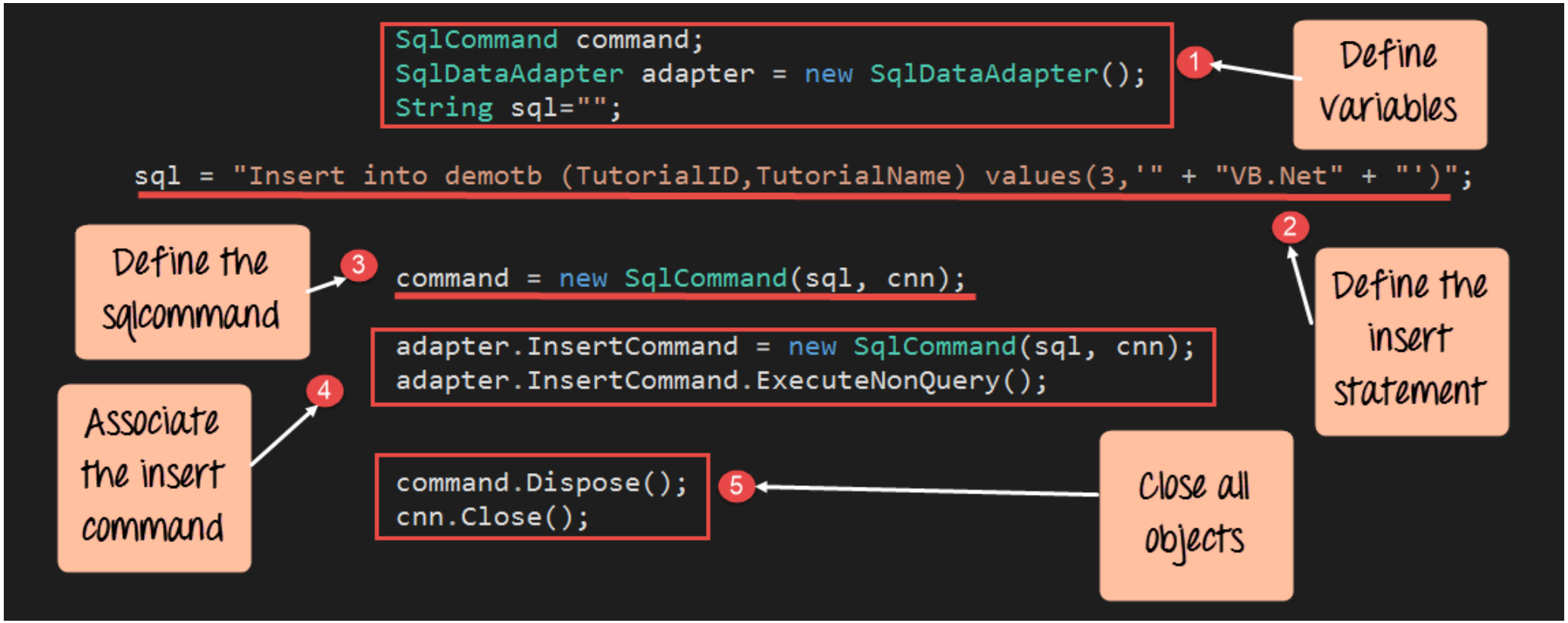
- Just like Accessing data, C# has the ability to insert records into the database as well. To showcase how to insert records into our database, let's take the same table structure which was used above.

TutorialID	TutorialName
1	C#
2	ASP.Net

Let's change the code in our form, so that we can insert the following row into the table

TutorialID	TutorialName
3	VB.Net

C# INSERT INTO DATABASE



C# UPDATE DATABASE

TutorialID	TutorialName
1	C#
2	ASP.Net
3	VB.Net

Let's change the code in our form, so that we can update the following row. The old row value is TutorialID as "3" and Tutorial Name as "VB.Net". Which we will update it to "VB.Net complete" while the row value for Tutorial ID will remain same.

Old row

TutorialID	TutorialName
3	VB.Net

New row

TutorialID	TutorialName
3	VB.Net complete

DELETING RECORDS

TutorialID	TutorialName
1	C#
2	ASP.Net
3	VB.Net complete

Let's change the code in our form, so that we can delete the following row

TutorialID	TutorialName
3	VB.Net complete

DELETING RECORDS