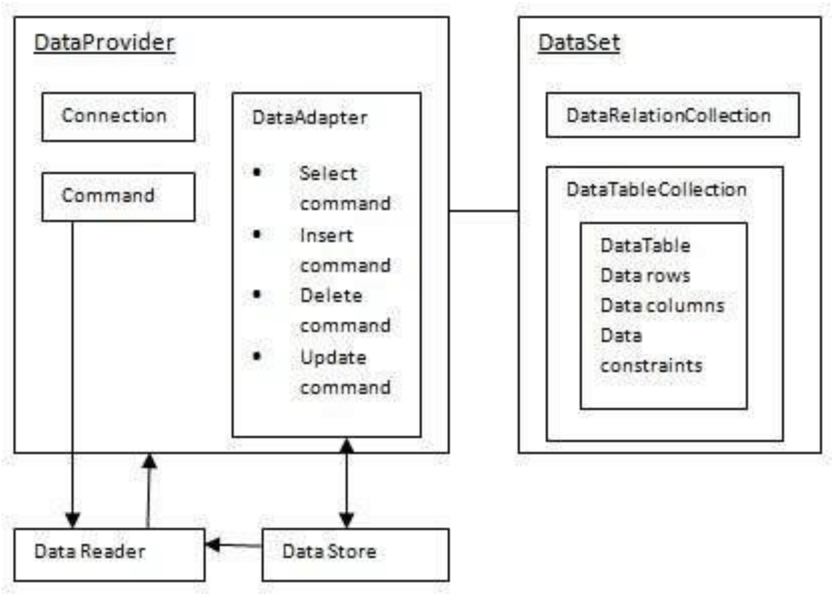
VB.Net – Data Access

Applications communicate with a database, firstly, to retrieve the data stored there and present it in a user-friendly way, and secondly, to update the database by inserting, modifying and deleting data.

Microsoft ActiveX Data Objects.Net (ADO.Net) is a model, a part of the .Net framework that is used by the .Net applications for retrieving, accessing and updating data.

ADO.Net Object Model

ADO.Net object model is nothing but the structured process flow through various components. The object model can be pictorially described as −



The data residing in a data store or database is retrieved through the **data provider**. Various components of the data provider retrieve data for the application and update data.

An application accesses data either through a dataset or a data reader.

- **Datasets** store data in a disconnected cache and the application retrieves data from it.
- **Data readers** provide data to the application in a read-only and forward-only mode.

Data Provider

A data provider is used for connecting to a database, executing commands and retrieving data, storing it in a dataset, reading the retrieved data and updating the database.

The data provider in ADO.Net consists of the following four objects −

Sr.No.	Objects & Description
1	**Connection** This component is used to set up a connection with a data source.
2	**Command** A command is a SQL statement or a stored procedure used to retrieve, insert, delete or modify data in a data source.
3	**DataReader** Data reader is used to retrieve data from a data source in a read-only and forward-only mode.
4	**DataAdapter** This is integral to the working of ADO.Net since data is transferred to and from a database through a data adapter. It retrieves data from a database into a dataset and updates the database. When changes are made to the dataset, the changes in the database are actually done by the data adapter.

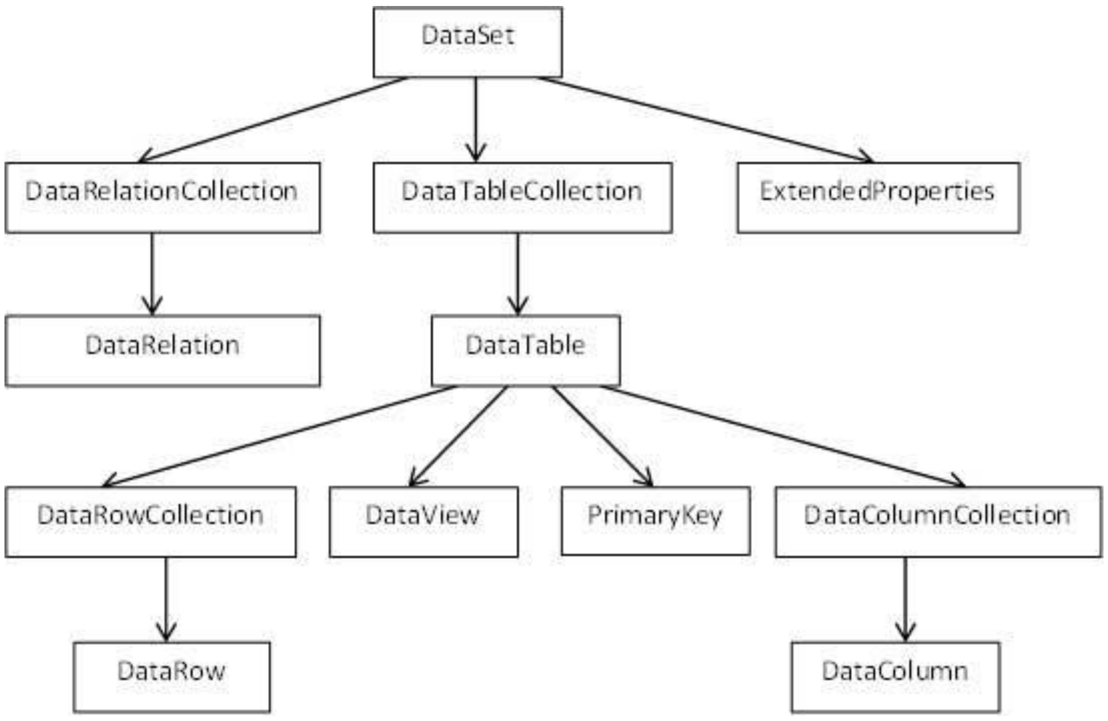
There are following different types of data providers included in ADO.Net

- The .Net Framework data provider for SQL Server - provides access to Microsoft SQL Server.
- The .Net Framework data provider for OLE DB - provides access to data sources exposed by using OLE DB.
- The .Net Framework data provider for ODBC - provides access to data sources exposed by ODBC.
- The .Net Framework data provider for Oracle - provides access to Oracle data source.
- The EntityClient provider - enables accessing data through Entity Data Model (EDM) applications.

DataSet

DataSet is an in-memory representation of data. It is a disconnected, cached set of records that are retrieved from a database. When a connection is established with the database, the data adapter creates a dataset and stores data in it. After the data is retrieved and stored in a dataset, the connection with the database is closed. This is called the 'disconnected architecture'. The dataset works as a virtual database containing tables, rows, and columns.

The following diagram shows the dataset object model −



The DataSet class is present in the **System.Data** namespace. The following table describes all the components of DataSet −

Sr.No.	Components & Description
1	**DataTableCollection** It contains all the tables retrieved from the data source.
2	**DataRelationCollection** It contains relationships and the links between tables in a data set.
3	**ExtendedProperties** It contains additional information, like the SQL statement for retrieving data, time of retrieval, etc.
4	**DataTable** It represents a table in the DataTableCollection of a dataset. It consists of the DataRow and DataColumn objects. The DataTable objects are case-sensitive.
5	**DataRelation**

	It represents a relationship in the DataRelationshipCollection of the dataset. It is used to relate two DataTable objects to each other through the DataColumn objects.
6	**DataRowCollection** It contains all the rows in a DataTable.
7	**DataView** It represents a fixed customized view of a DataTable for sorting, filtering, searching, editing and navigation.
8	**PrimaryKey** It represents the column that uniquely identifies a row in a DataTable.
9	**DataRow** It represents a row in the DataTable. The DataRow object and its properties and methods are used to retrieve, evaluate, insert, delete, and update values in the DataTable. The NewRow method is used to create a new row and the Add method adds a row to the table.
10	**DataColumnCollection** It represents all the columns in a DataTable.
11	**DataColumn** It consists of the number of columns that comprise a DataTable.

Connecting to a Database

The .Net Framework provides two types of Connection classes −

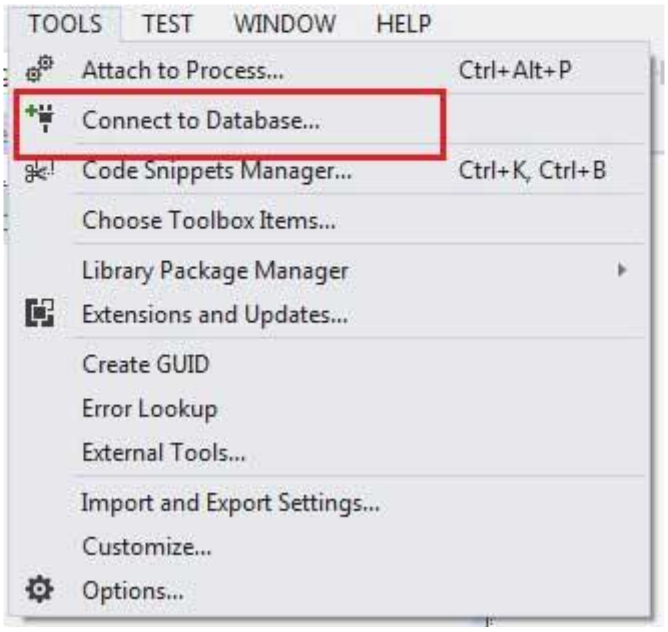
- **SqlConnection** − designed for connecting to Microsoft SQL Server.
- **OleDbConnection** − designed for connecting to a wide range of databases, like Microsoft Access and Oracle.

Example 1

We have a table stored in Microsoft SQL Server, named Customers, in a database named testDB. Please consult 'SQL Server' tutorial for creating databases and database tables in SQL Server.

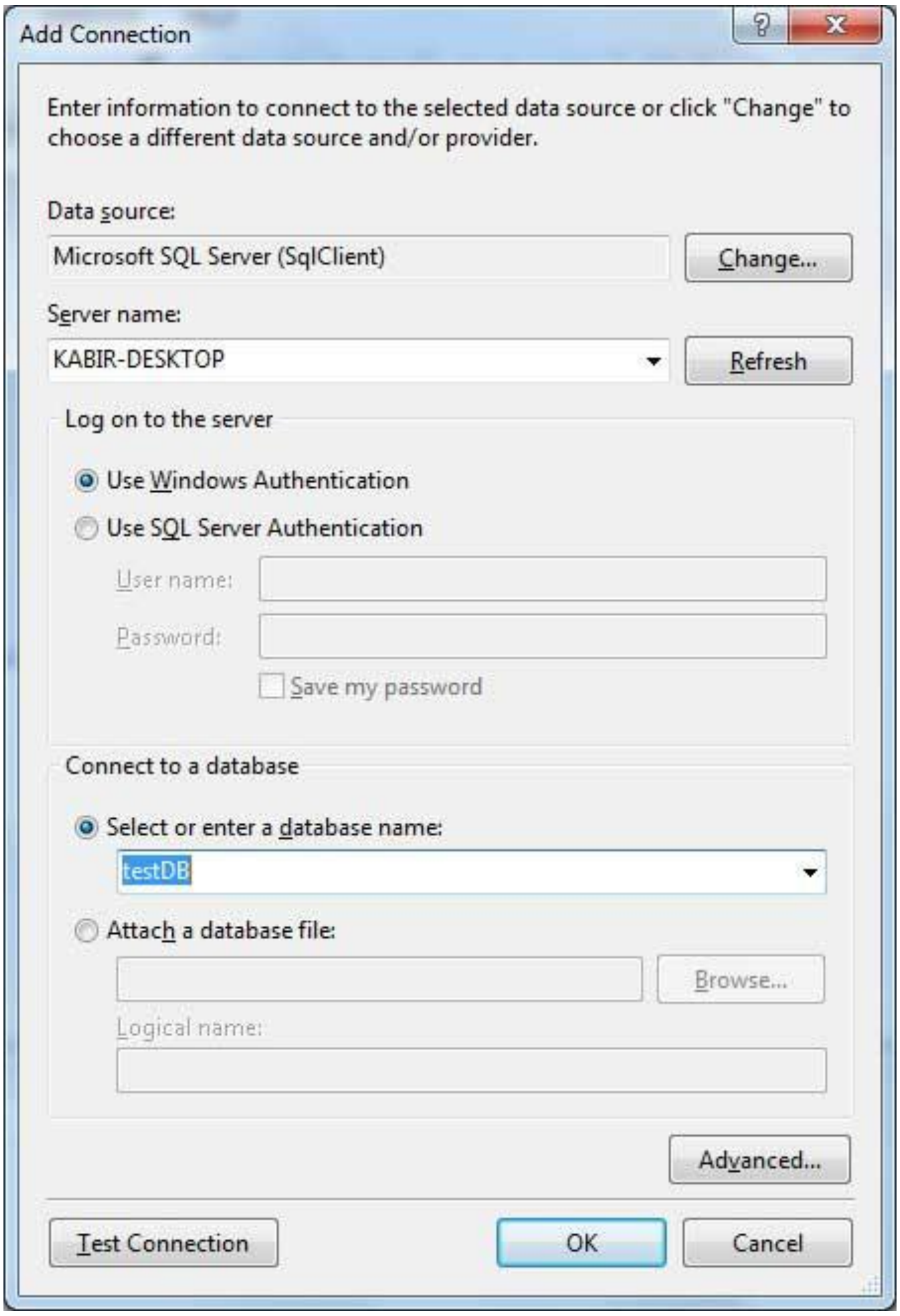
Let us connect to this database. Take the following steps −

- Select TOOLS → Connect to Database



- Select a server name and the database name in the Add Connection dialog box.

M


Add Connection

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source: Microsoft SQL Server (SqlClient)

Server name: KABIR-DESKTOP

Log on to the server

- [✓ Use Windows Authentication]
- Use SQL Server Authentication

User name:

Password:

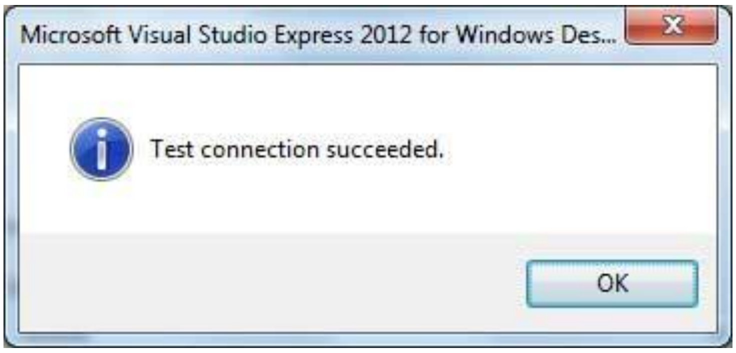
Save my password

Connect to a database

- [✓ Select or enter a database name:] testDB
- Attach a database file:

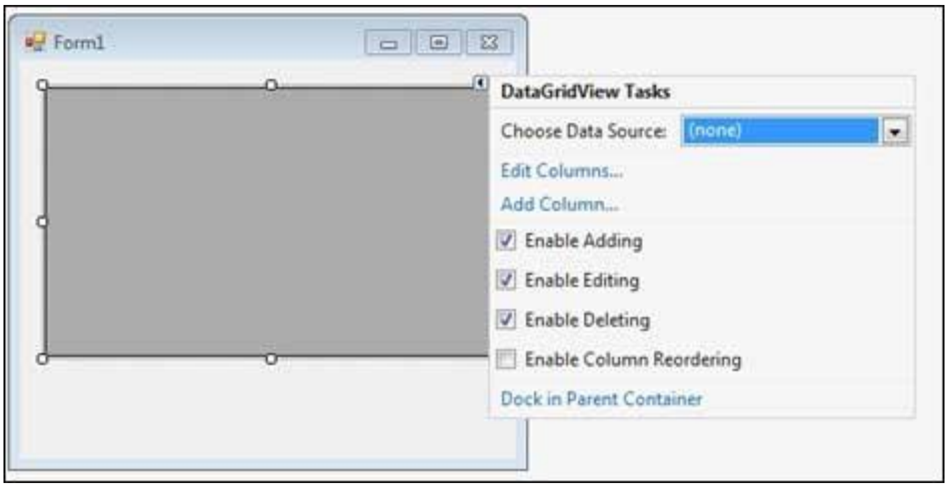
Logical name:

- Click on the Test Connection button to check if the connection succeeded.

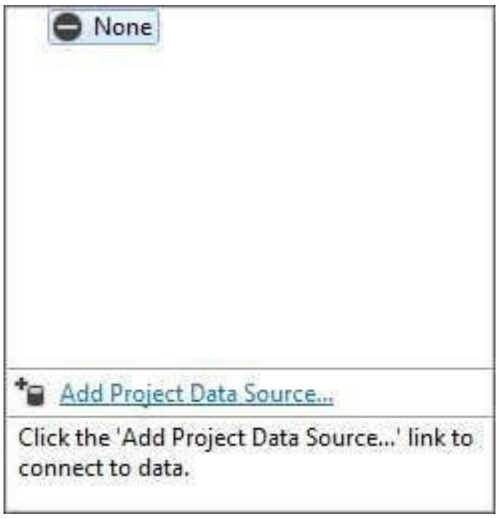

Microsoft Visual Studio Express 2012 for Windows Des...

Test connection succeeded.

- Add a DataGridView on the form.

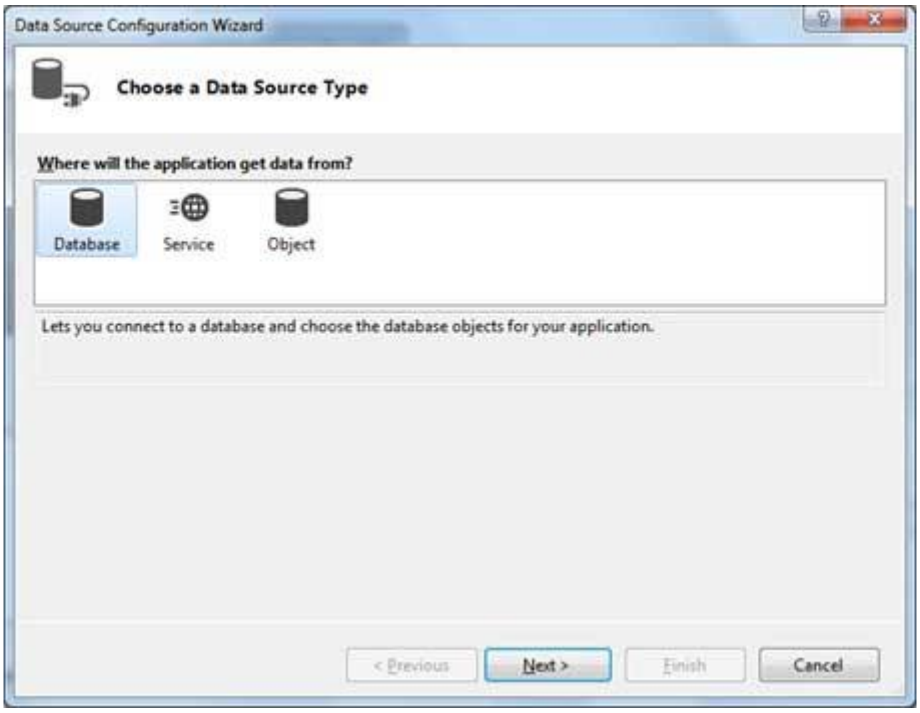


- Click on the Choose Data Source combo box.
- Click on the Add Project Data Source link.



Click the 'Add Project Data Source...' link to connect to data.

- This opens the Data Source Configuration Wizard.
- Select Database as the data source type

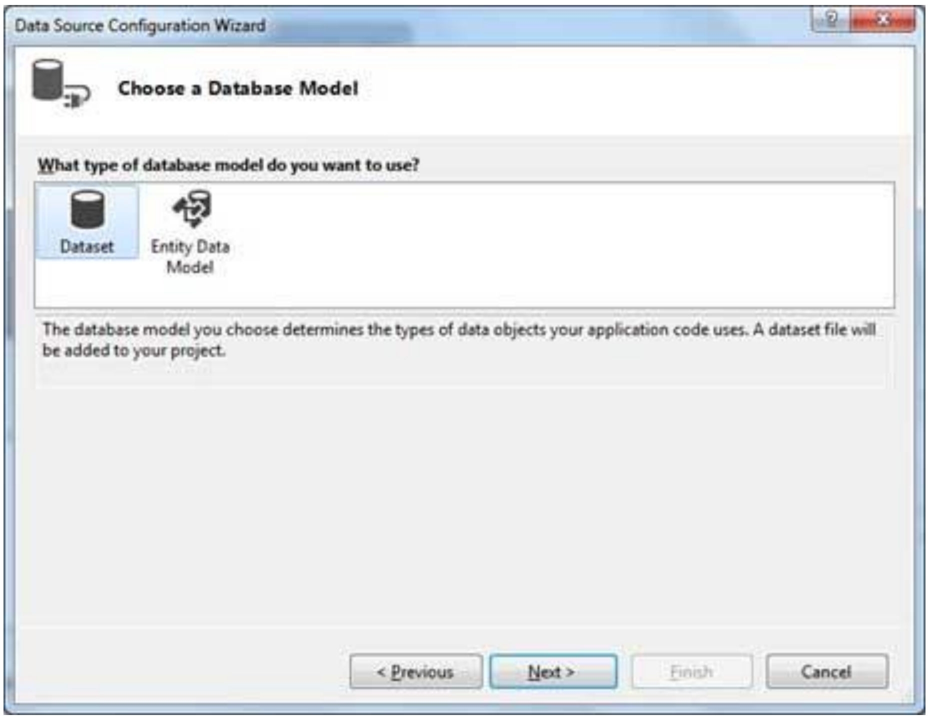

Data Source Configuration Wizard

Choose a Data Source Type

Where will the application get data from?

Lets you connect to a database and choose the database objects for your application.

- Choose DataSet as the database model.

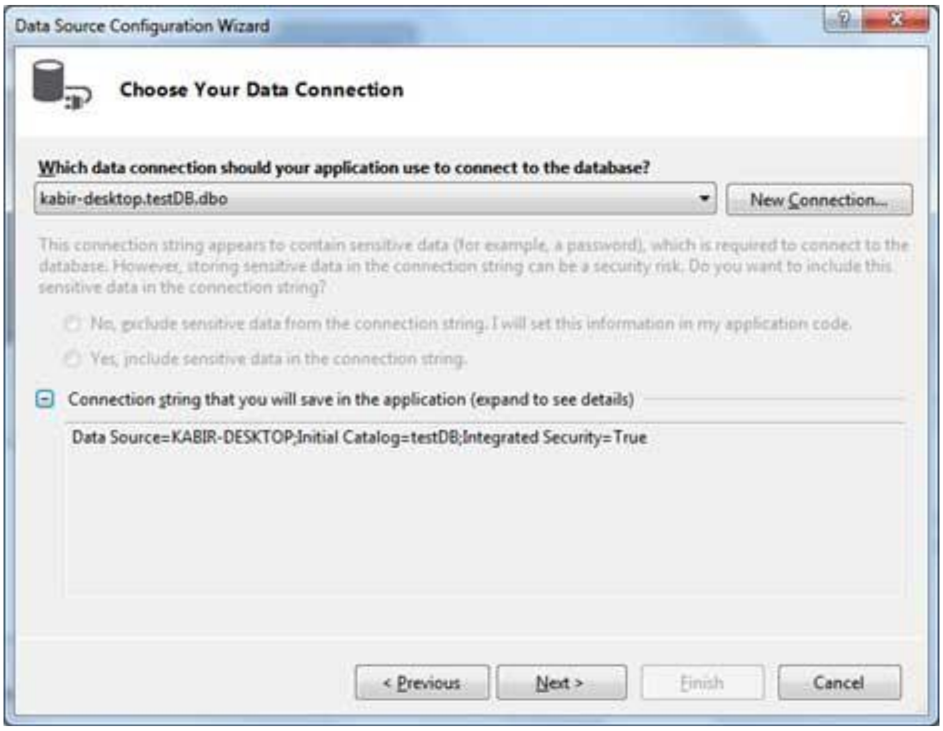

Data Source Configuration Wizard

Choose a Database Model

What type of database model do you want to use?

The database model you choose determines the types of data objects your application code uses. A dataset file will be added to your project.

- Choose the connection already set up.


Data Source Configuration Wizard

Choose Your Data Connection

Which data connection should your application use to connect to the database?

kabir-desktop.testDB.dbo

This connection string appears to contain sensitive data (for example, a password), which is required to connect to the database. However, storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

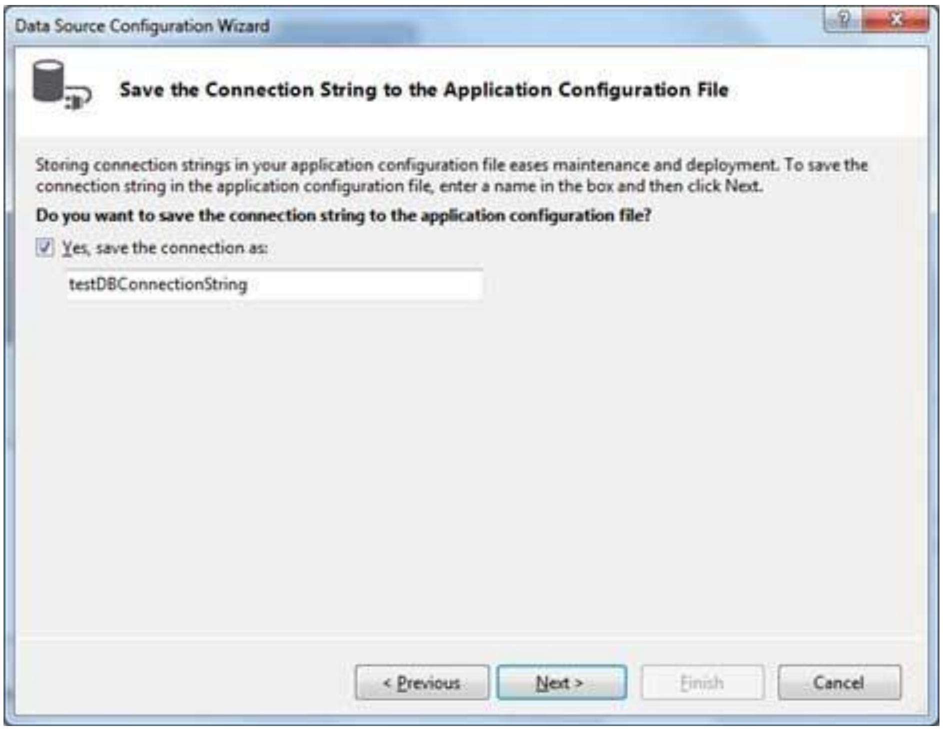
No, exclude sensitive data from the connection string. I will set this information in my application code.

Yes, include sensitive data in the connection string.

Connection string that you will save in the application (expand to see details)

Data Source=KABIR-DESKTOP;Initial Catalog=testDB;Integrated Security=True

- Save the connection string.


Data Source Configuration Wizard

Save the Connection String to the Application Configuration File

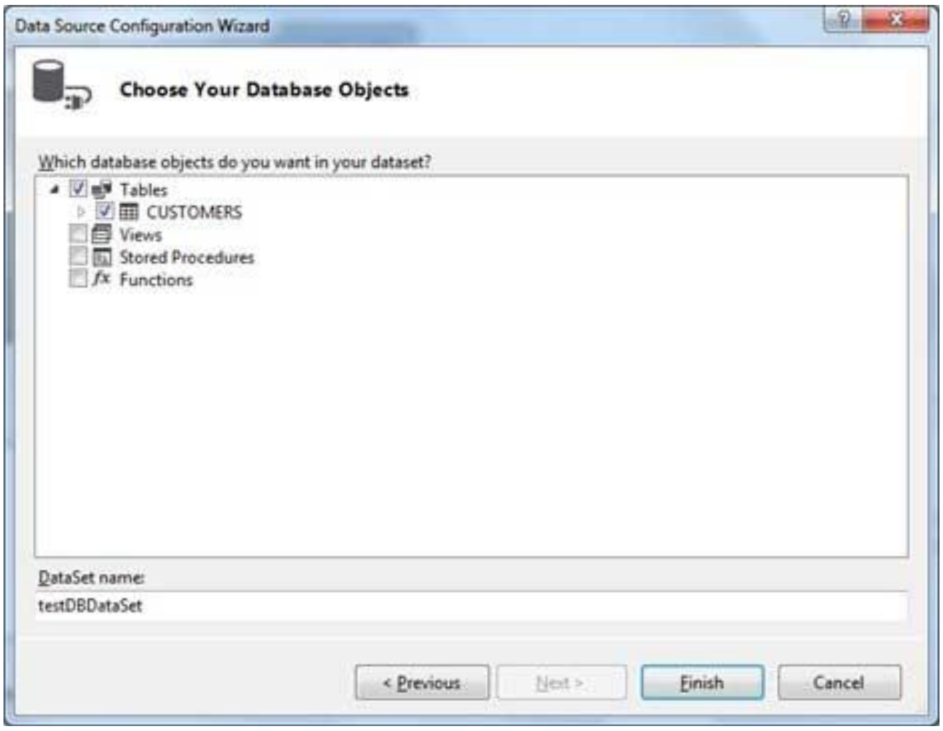
Storing connection strings in your application configuration file eases maintenance and deployment. To save the connection string in the application configuration file, enter a name in the box and then click Next.

Do you want to save the connection string to the application configuration file?

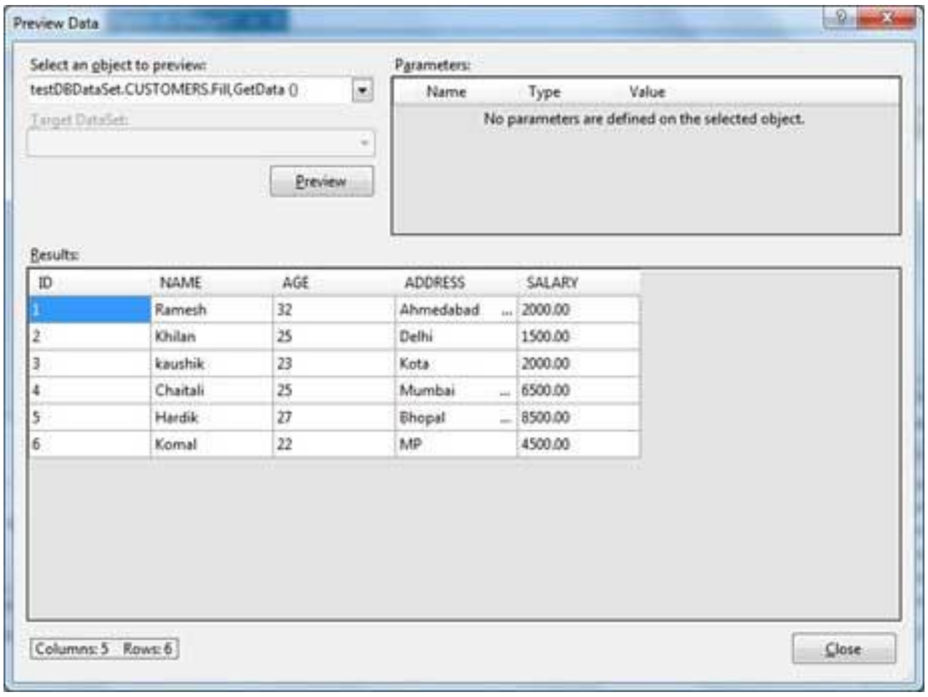
☑ Yes, save the connection as:

testDBConnectionString

- Choose the database object, Customers table in our example, and click the Finish button.

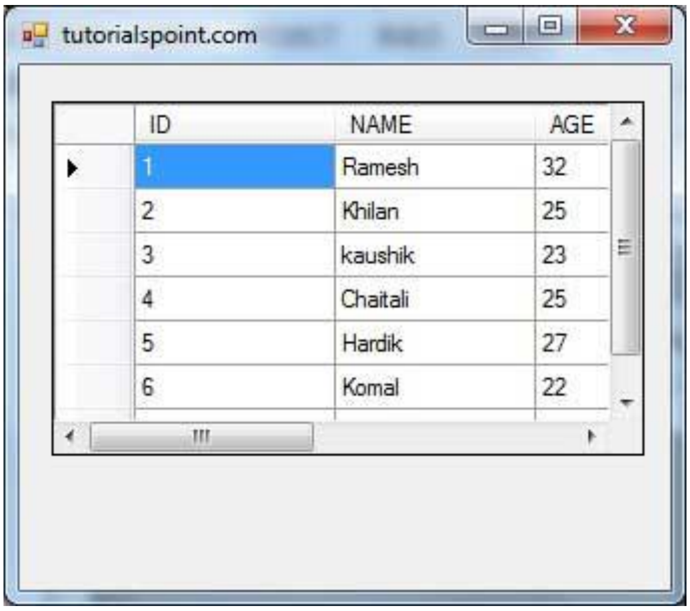

Data Source Configuration Wizard
Choose Your Database Objects
Which database objects do you want in your dataset?
Tables
CUSTOMERS
Views
Stored Procedures
Functions
DataSet name:
testDBDataSet
< Previous
Next >
Finish
Cancel

- Select the Preview Data link to see the data in the Results grid −


Preview Data
Select an object to preview:
testDBDataSet.CUSTOMERS.Fill,GetData ()
Target DataSet:
Preview
Parameters:
Name
Type
Value
No parameters are defined on the selected object.
Results:
Columns: 5 Rows: 6
Close

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad ...	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai ...	6500.00
5	Hardik	27	Bhopal ...	8500.00
6	Komal	22	MP	4500.00

When the application is run using **Start** button available at the Microsoft Visual Studio tool bar, it will show the following window −



ID	NAME	AGE
1	Ramesh	32
2	Khilan	25
3	kaushik	23
4	Chaitali	25
5	Hardik	27
6	Komal	22

Example 2

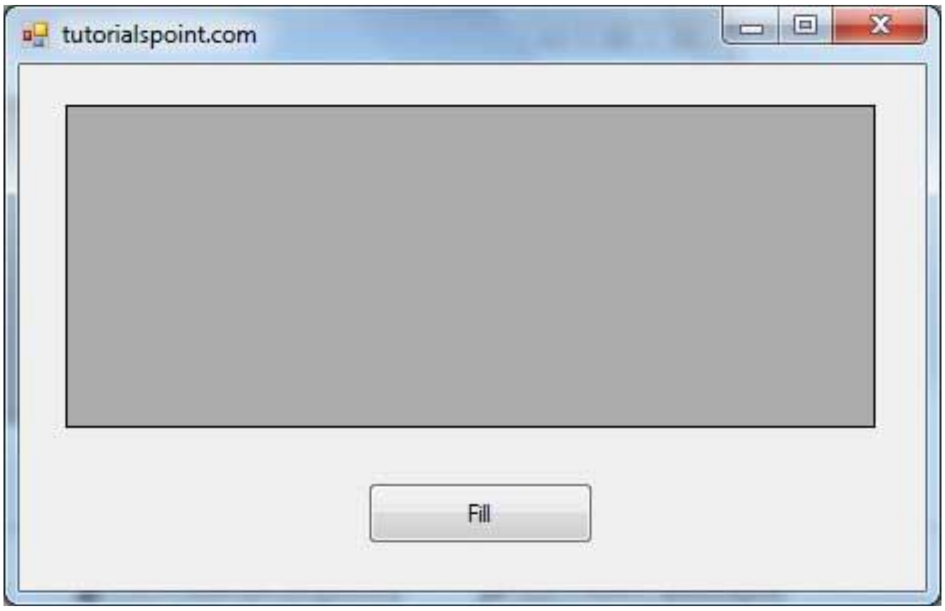
In this example, let us access data in a DataGridView control using code. Take the following steps −

- Add a DataGridView control and a button in the form.
- Change the text of the button control to 'Fill'.
- Double click the button control to add the required code for the Click event of the button, as shown below −

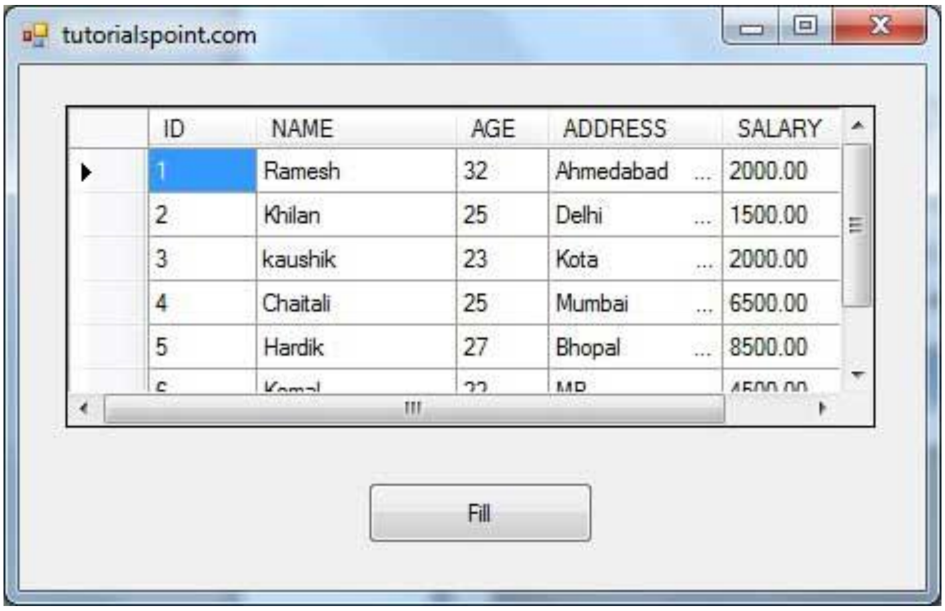
```
Imports System.Data.SqlClient
Public Class Form1
   Private Sub Form1_Load(sender As Object, e As EventArgs) _
   Handles MyBase.Load
      'TODO: This line of code loads data into the
'TestDBDataSet.CUSTOMERS' table.
      You can move, or remove it, as needed.

      Me.CUSTOMERSTableAdapter.Fill(Me.TestDBDataSet.CUSTOMERS)
      ' Set the caption bar text of the form.
      Me.Text = "tutorialspoint.com"
   End Sub

   Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click
      Dim connection As SqlConnection = New sqlconnection()
      connection.ConnectionString = "Data Source=KABIR-DESKTOP; _
         Initial Catalog=testDB;Integrated Security=True"
      connection.Open()
      Dim adp As SqlDataAdapter = New SqlDataAdapter _
      ("select * from Customers", connection)
      Dim ds As DataSet = New DataSet()
      adp.Fill(ds)
```

```
        DataGridView1.DataSource = ds.Tables(0)
    End Sub
End Class
```

- When the above code is executed and run using **Start** button available at the Microsoft Visual Studio tool bar, it will show the following window −



- Clicking the Fill button displays the table on the data grid view control −



	ID	NAME	AGE	ADDRESS	SALARY
▶	1	Ramesh	32	Ahmedabad ...	2000.00
	2	Khilan	25	Delhi ...	1500.00
	3	kaushik	23	Kota ...	2000.00
	4	Chaitali	25	Mumbai ...	6500.00
	5	Hardik	27	Bhopal ...	8500.00
	6	Komal	22	MP	4500.00

Creating Table, Columns and Rows

We have discussed that the DataSet components like DataTable, DataColumn and DataRow allow us to create tables, columns and rows, respectively.

The following example demonstrates the concept −

Example 3

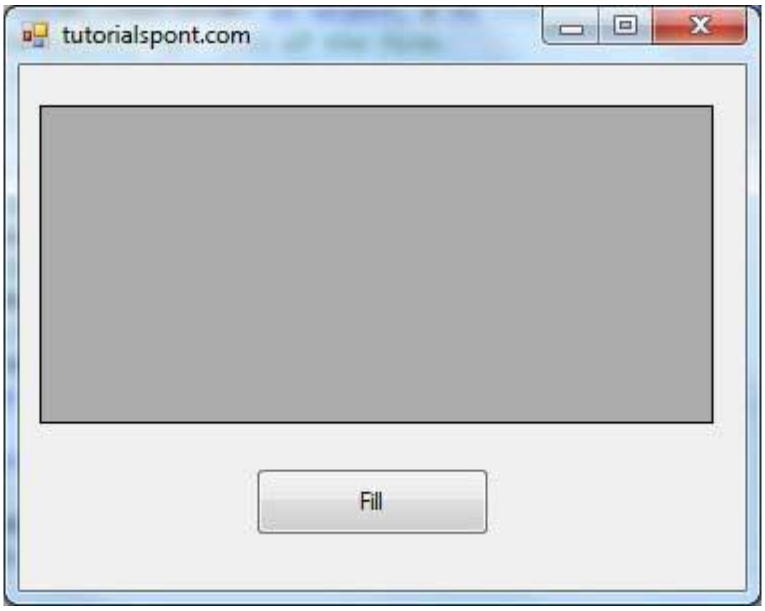
So far, we have used tables and databases already existing in our computer. In this example, we will create a table, add columns, rows and data into it and display the table using a DataGridView object.

Take the following steps −

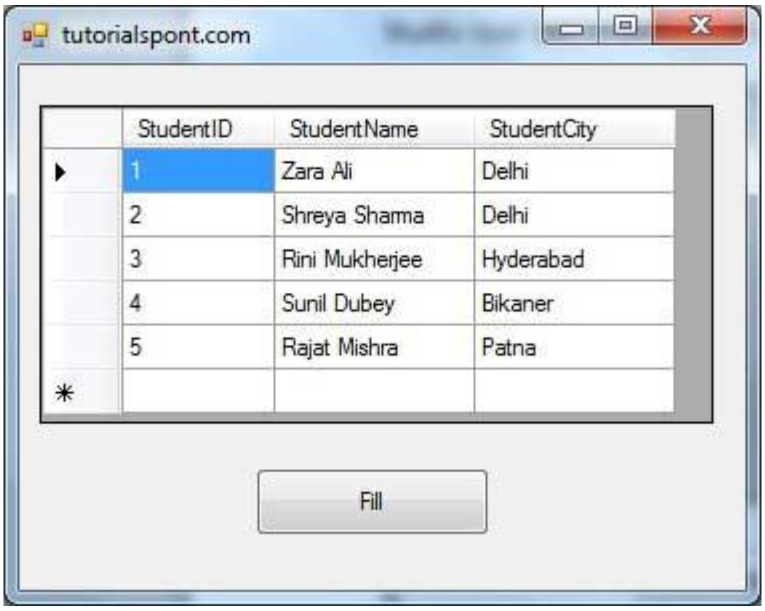
- Add a DataGridView control and a button in the form.
- Change the text of the button control to 'Fill'.
- Add the following code in the code editor.

```
Public Class Form1
   Private Sub Form1_Load(sender As Object, e As EventArgs)
Handles MyBase.Load
      ' Set the caption bar text of the form.
      Me.Text = "tutorialspont.com"
   End Sub

   Private Function CreateDataSet() As DataSet
      'creating a DataSet object for tables
      Dim dataset As DataSet = New DataSet()
      ' creating the student table
      Dim Students As DataTable = CreateStudentTable()
      dataset.Tables.Add(Students)
      Return dataset
   End Function

   Private Function CreateStudentTable() As DataTable
      Dim Students As DataTable
      Students = New DataTable("Student")
      ' adding columns
      AddNewColumn(Students, "System.Int32", "StudentID")
      AddNewColumn(Students, "System.String", "StudentName")
      AddNewColumn(Students, "System.String", "StudentCity")
      ' adding rows
      AddNewRow(Students, 1, "Zara Ali", "Kolkata")
      AddNewRow(Students, 2, "Shreya Sharma", "Delhi")
      AddNewRow(Students, 3, "Rini Mukherjee", "Hyderabad")
      AddNewRow(Students, 4, "Sunil Dubey", "Bikaner")
      AddNewRow(Students, 5, "Rajat Mishra", "Patna")
      Return Students
   End Function

   Private Sub AddNewColumn(ByRef table As DataTable, _
   ByVal columnType As String, ByVal columnName As String)
      Dim column As DataColumn = _
         table.Columns.Add(columnName, Type.GetType(columnType))
   End Sub

   'adding data into the table
   Private Sub AddNewRow(ByRef table As DataTable, ByRef id As
Integer,_
   ByRef name As String, ByRef city As String)
      Dim newrow As DataRow = table.NewRow()
      newrow("StudentID") = id
      newrow("StudentName") = name
      newrow("StudentCity") = city
      table.Rows.Add(newrow)
   End Sub

   Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click
      Dim ds As New DataSet
      ds = CreateDataSet()
      DataGridView1.DataSource = ds.Tables("Student")
   End Sub
End Class
```

- When the above code is executed and run using **Start** button available at the Microsoft Visual Studio tool bar, it will show the following window −



- Clicking the Fill button displays the table on the data grid view control −



StudentID	StudentName	StudentCity
1	Zara Ali	Delhi
2	Shreya Sharma	Delhi
3	Rini Mukherjee	Hyderabad
4	Sunil Dubey	Bikaner
5	Rajat Mishra	Patna

VB.Net - Excel Sheet

VB.Net provides support for interoperability between the COM object model of Microsoft Excel 2010 and your application.

To avail this interoperability in your application, you need to import the namespace **Microsoft.Office.Interop.Excel** in your Windows Form Application.

Creating an Excel Application from VB.Net

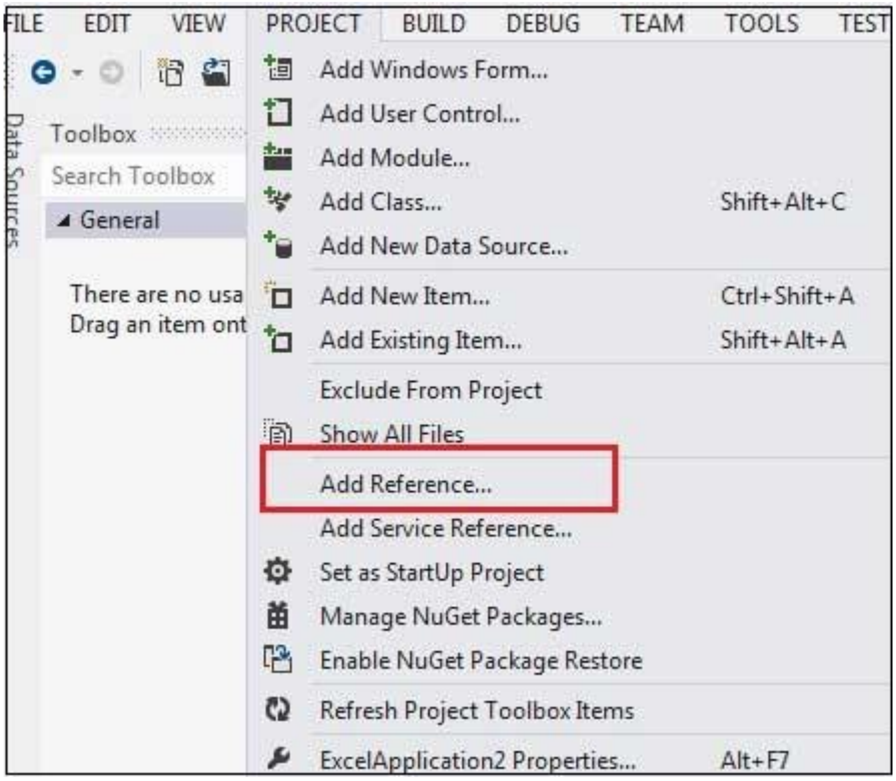
Let's start with creating a Window Forms Application by following the following steps in Microsoft Visual Studio: **File** → **New Project** → **Windows Forms Applications**

Finally, select OK, Microsoft Visual Studio creates your project and displays following **Form1**.

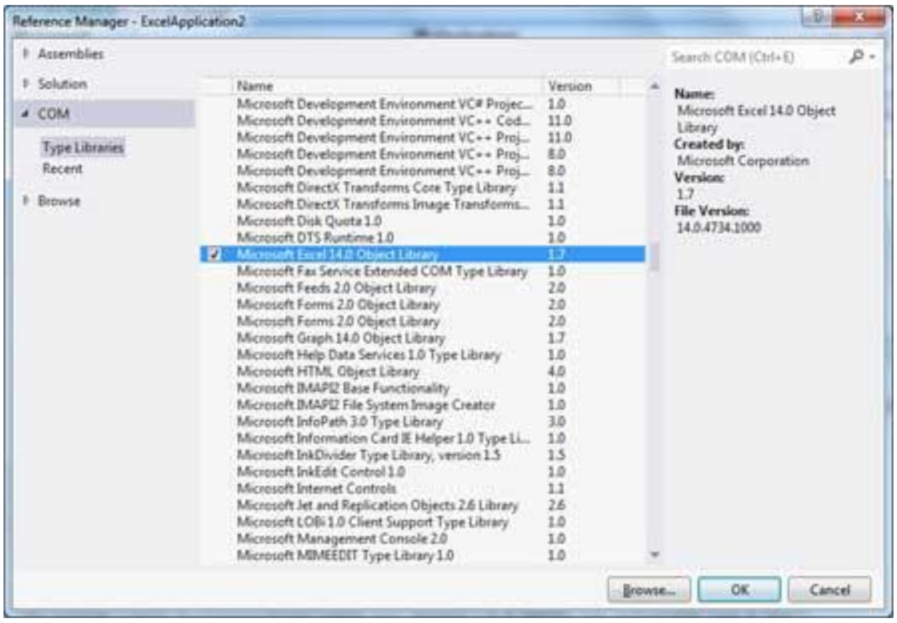
Insert a Button control Button1 in the form.

Add a reference to Microsoft Excel Object Library to your project. To do this −

- Select Add Reference from the Project Menu.



- On the COM tab, locate Microsoft Excel Object Library and then click Select.


Reference Manager - ExcelApplication2

- Assemblies
- Solution
- COM
  - Type Libraries
  - Recent
- Browse

Search COM (Ctrl+E)

	Name	Version
	Microsoft Development Environment VC# Projec...	1.0
	Microsoft Development Environment VC++ Cod...	11.0
	Microsoft Development Environment VC++ Proj...	11.0
	Microsoft Development Environment VC++ Proj...	8.0
	Microsoft Development Environment VC++ Proj...	8.0
	Microsoft DirectX Transforms Core Type Library	1.1
	Microsoft DirectX Transforms Image Transforms...	1.1
	Microsoft Disk Quota 1.0	1.0
	Microsoft DTS Runtime 1.0	1.0
✓	Microsoft Excel 14.0 Object Library	1.7
	Microsoft Fax Service Extended COM Type Library	1.0
	Microsoft Feeds 2.0 Object Library	2.0
	Microsoft Forms 2.0 Object Library	2.0
	Microsoft Forms 2.0 Object Library	2.0
	Microsoft Graph 14.0 Object Library	1.7
	Microsoft Help Data Services 1.0 Type Library	1.0
	Microsoft HTML Object Library	4.0
	Microsoft IMAPI2 Base Functionality	1.0
	Microsoft IMAPI2 File System Image Creator	1.0
	Microsoft InfoPath 3.0 Type Library	3.0
	Microsoft Information Card IE Helper 1.0 Type Li...	1.0
	Microsoft InkDivider Type Library, version 1.5	1.5
	Microsoft InkEdit Control 1.0	1.0
	Microsoft Internet Controls	1.1
	Microsoft Jet and Replication Objects 2.6 Library	2.6
	Microsoft LOBi 1.0 Client Support Type Library	1.0
	Microsoft Management Console 2.0	1.0
	Microsoft MIMEEDIT Type Library 1.0	1.0

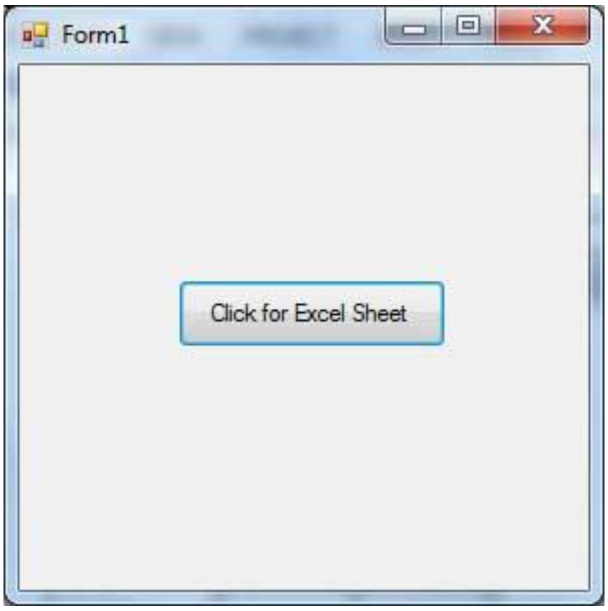
Name:
Microsoft Excel 14.0 Object Library
Created by:
Microsoft Corporation
Version:
1.7
File Version:
14.0.4734.1000

- Click OK.

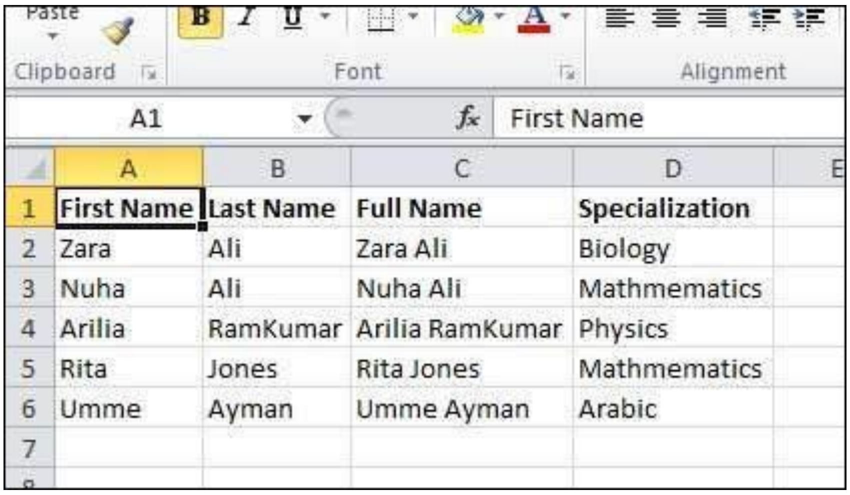
Double click the code window and populate the Click event of Button1, as shown below.

```
'  Add the following code snippet on top of Form1.vb
Imports Excel = Microsoft.Office.Interop.Excel
Public Class Form1
   Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click
      Dim appXL As Excel.Application
      Dim wbXl As Excel.Workbook
      Dim shXL As Excel.Worksheet
      Dim raXL As Excel.Range

      ' Start Excel and get Application object.
      appXL = CreateObject("Excel.Application")
      appXL.Visible = True

      ' Add a new workbook.
      wbXl = appXL.Workbooks.Add
      shXL = wbXl.ActiveSheet

      ' Add table headers going cell by cell.
      shXL.Cells(1, 1).Value = "First Name"
      shXL.Cells(1, 2).Value = "Last Name"
      shXL.Cells(1, 3).Value = "Full Name"
      shXL.Cells(1, 4).Value = "Specialization"

      ' Format A1:D1 as bold, vertical alignment = center.
      With shXL.Range("A1", "D1")
         .Font.Bold = True
         .VerticalAlignment = Excel.XlVAlign.xlVAlignCenter
      End With
      ' Create an array to set multiple values at once.
      Dim students(5, 2) As String
      students(0, 0) = "Zara"
      students(0, 1) = "Ali"
      students(1, 0) = "Nuha"
      students(1, 1) = "Ali"
      students(2, 0) = "Arilia"
      students(2, 1) = "RamKumar"
      students(3, 0) = "Rita"
      students(3, 1) = "Jones"
      students(4, 0) = "Umme"
      students(4, 1) = "Ayman"

      ' Fill A2:B6 with an array of values (First and Last
Names).
      shXL.Range("A2", "B6").Value = students

      ' Fill C2:C6 with a relative formula (=A2 & " " & B2).
      raXL = shXL.Range("C2", "C6")
      raXL.Formula = "=A2 & "" "" & B2"

      ' Fill D2:D6 values.
      With shXL
         .Cells(2, 4).Value = "Biology"
         .Cells(3, 4).Value = "Mathmematics"
         .Cells(4, 4).Value = "Physics"
         .Cells(5, 4).Value = "Mathmematics"
         .Cells(6, 4).Value = "Arabic"
      End With

      ' AutoFit columns A:D.
      raXL = shXL.Range("A1", "D1")
      raXL.EntireColumn.AutoFit()

      ' Make sure Excel is visible and give the user control
      ' of Excel's lifetime.
      appXL.Visible = True
      appXL.UserControl = True

      ' Release object references.
      raXL = Nothing
      shXL = Nothing
      wbXl = Nothing
      appXL.Quit()
      appXL = Nothing
      Exit Sub
Err_Handler:
      MsgBox(Err.Description, vbCritical, "Error: " & Err.Number)
   End Sub
End Class
```

When the above code is executed and run using **Start** button available at the Microsoft Visual Studio tool bar, it will show the following window −



Clicking on the Button would display the following excel sheet. You will be asked to save the workbook.



	A	B	C	D
1	**First Name**	**Last Name**	**Full Name**	**Specialization**
2	Zara	Ali	Zara Ali	Biology
3	Nuha	Ali	Nuha Ali	Mathmematics
4	Arilia	RamKumar	Arilia RamKumar	Physics
5	Rita	Jones	Rita Jones	Mathmematics
6	Umme	Ayman	Umme Ayman	Arabic
7				