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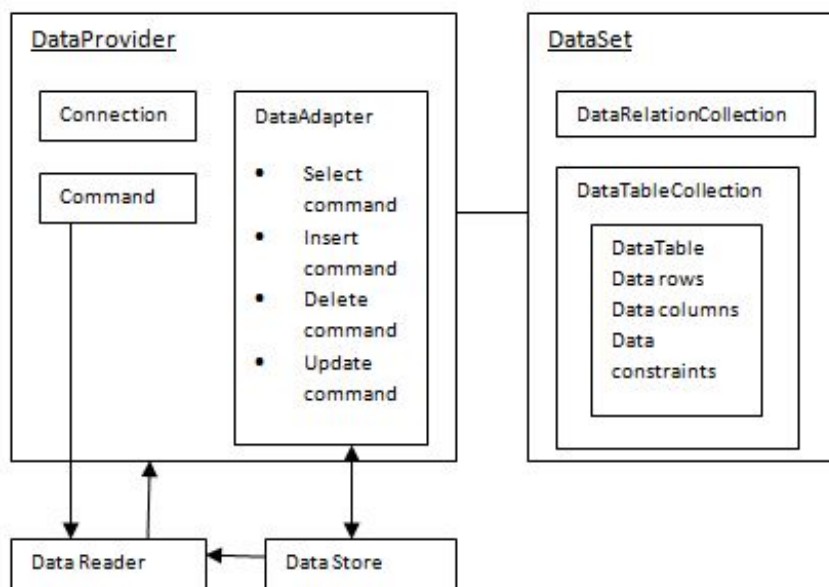
The data residing in a data store or database is retrieved through the **data provider**. Various components of the data provider retrieve data for the application and update data.

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:

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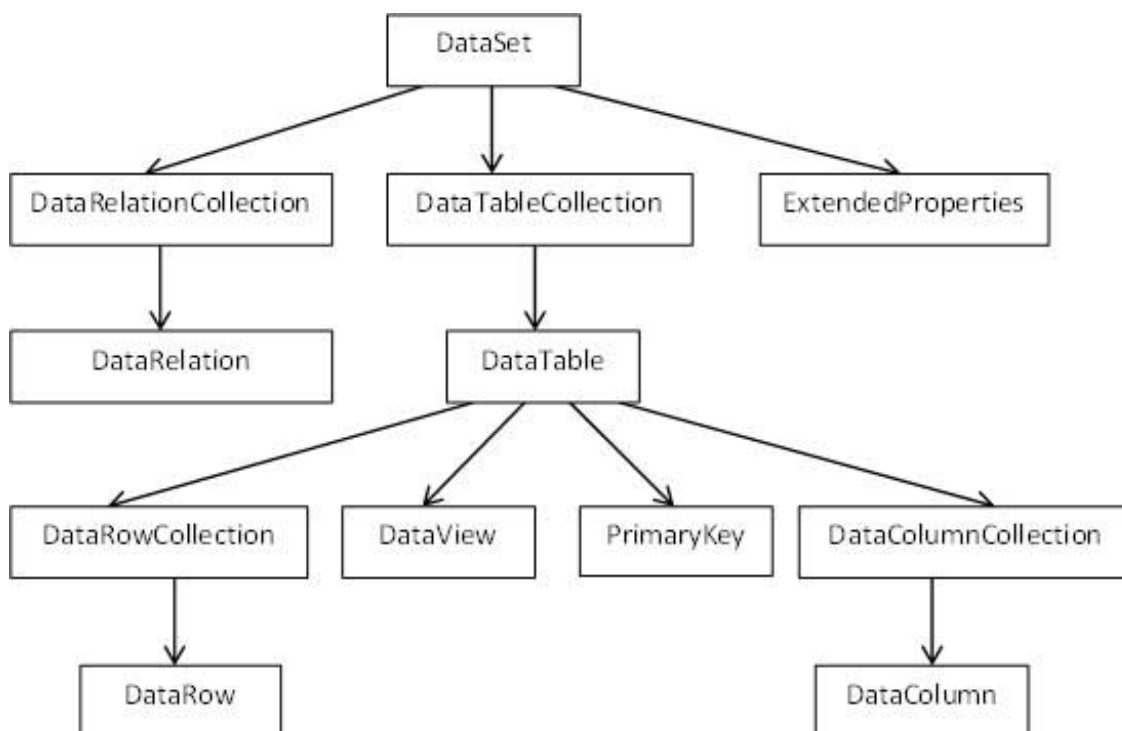
The DataSet class is present in the **System.Data** namespace. The following table describes all the components of DataSet:

S.N	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:
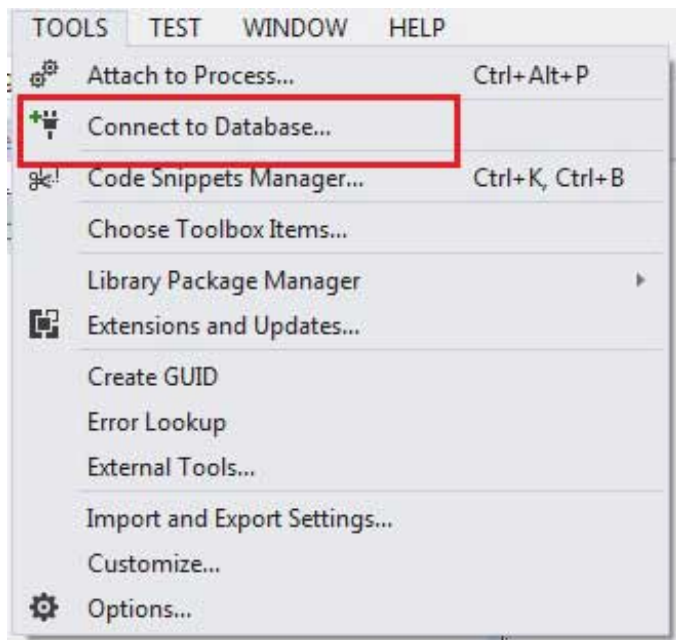
- **SqlConnection** - designed for connecting to Microsoft SQL Server.

- **OleDbConnection** - designed for connecting to a wide range of databases, like Microsoft Access and Oracle.
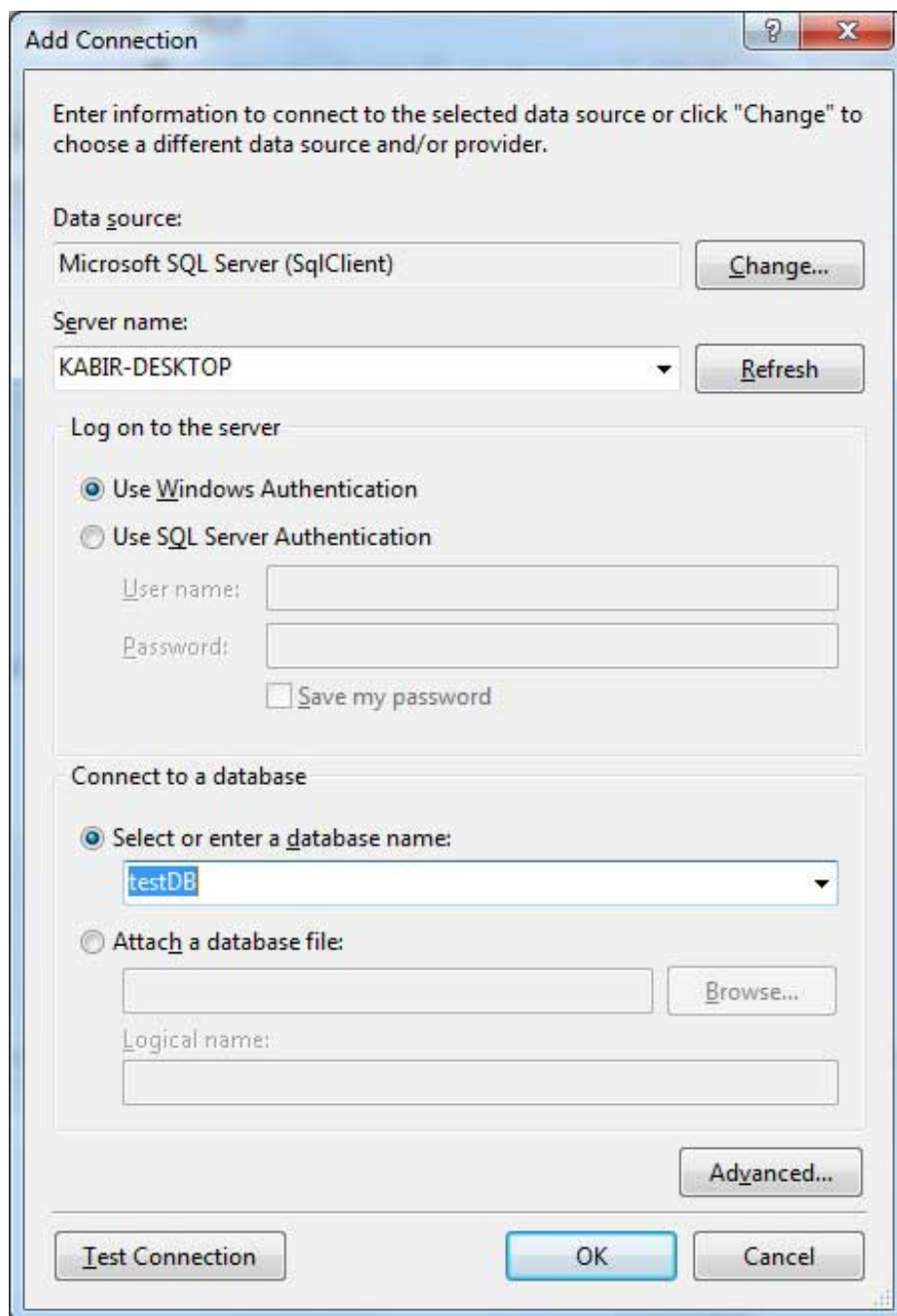
Example 1

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:

1. Select TOOLS -> Connect to Database

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

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)   Change...

Server name:

KABIR-DESKTOP ▼   Refresh

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:

     Browse...

Logical name:

Advanced...

Test Connection   OK   Cancel

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

Microsoft Visual Studio Express 2012 for Windows Des...
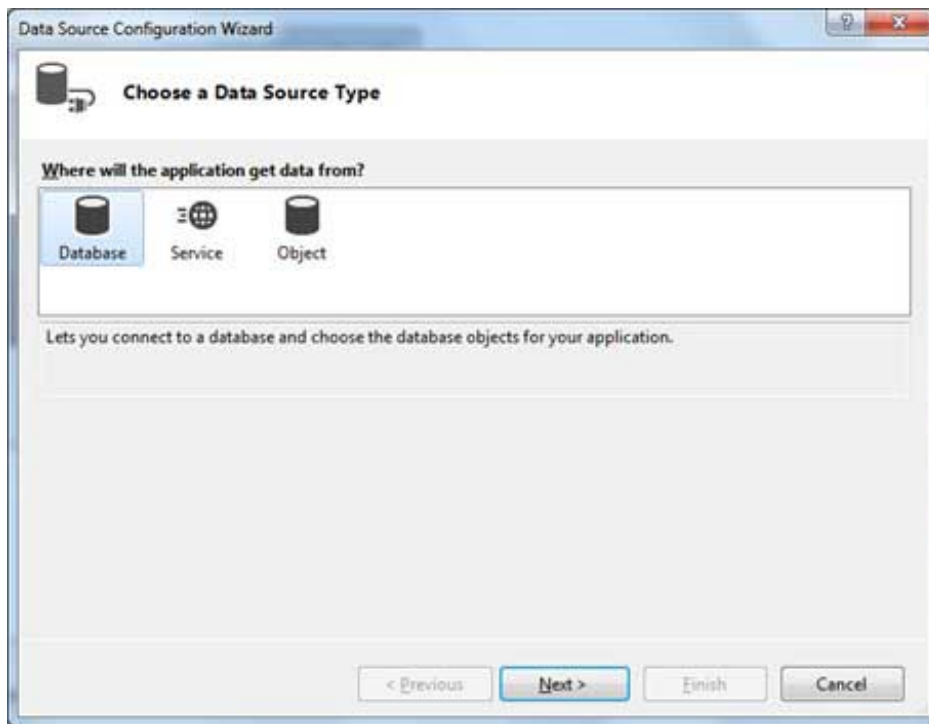
ⓘ Test connection succeeded.

OK

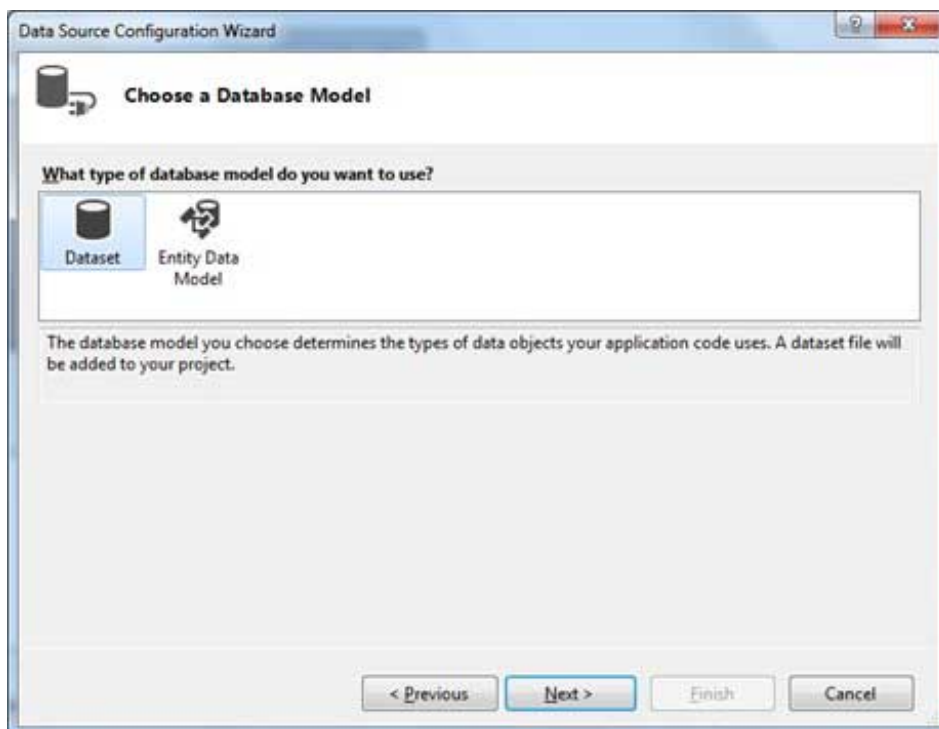4. Add a DataGridView on the form.
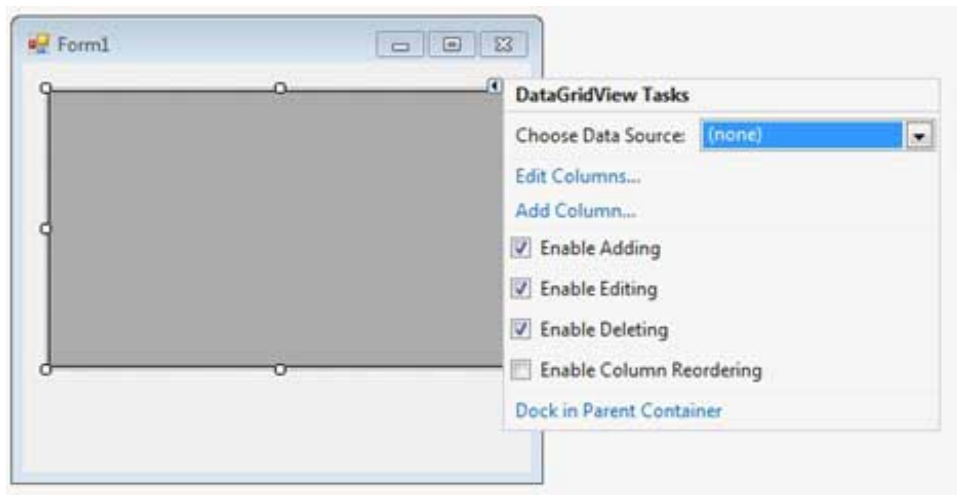
5. Click on the Choose Data Source combo box.

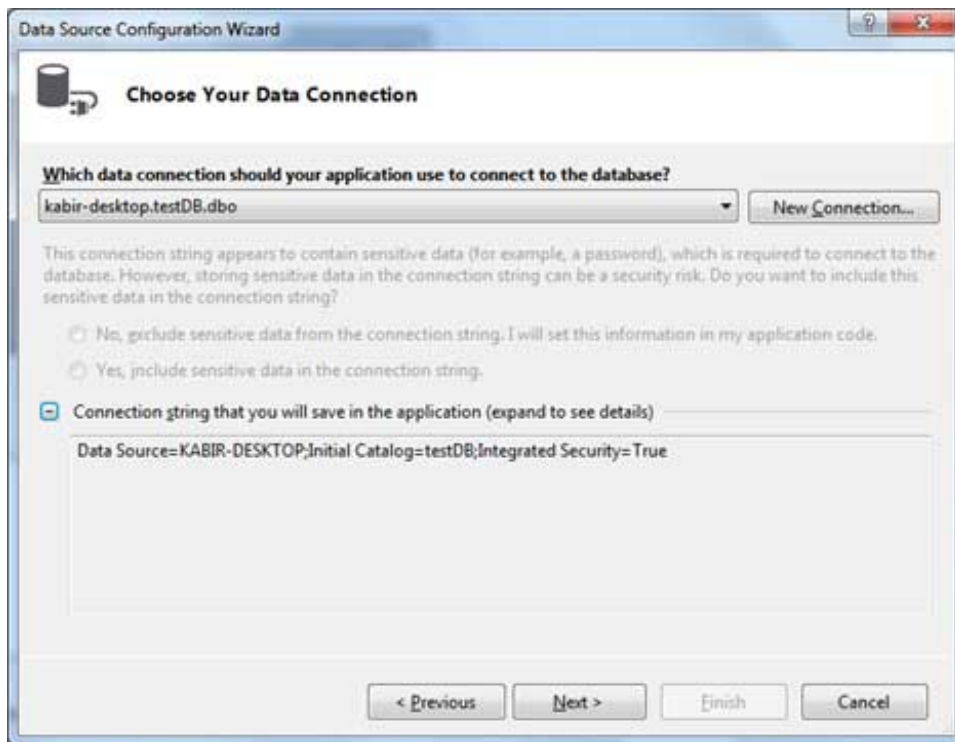6. Click on the Add Project Data Source link.
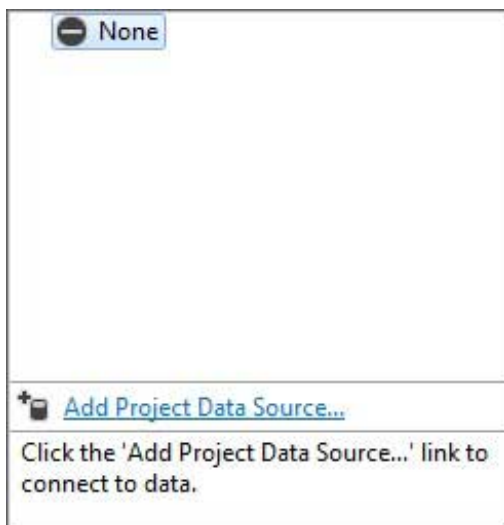
7. This opens the Data Source Configuration Wizard.
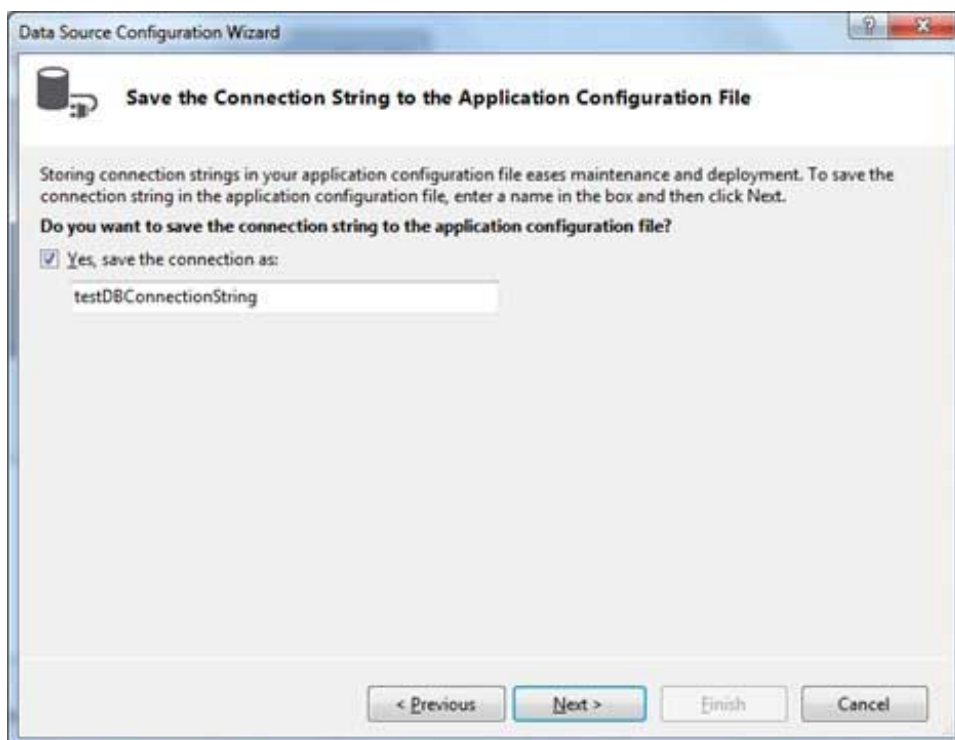
8. Select Database as the data source type

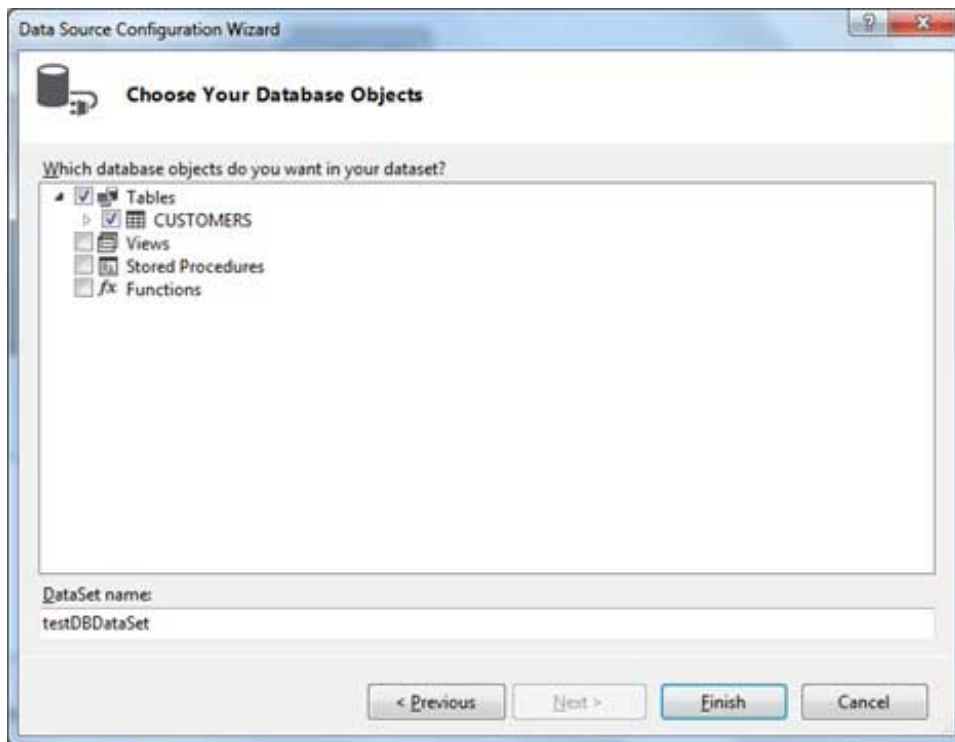9. Choose DataSet as the database model.

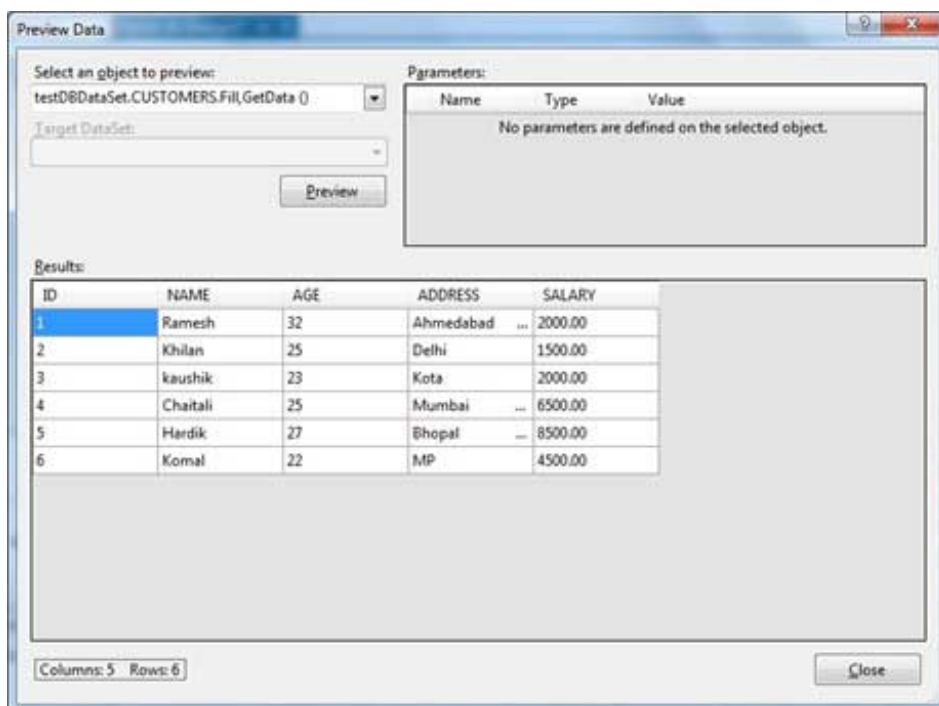10. Choose the connection already set up.

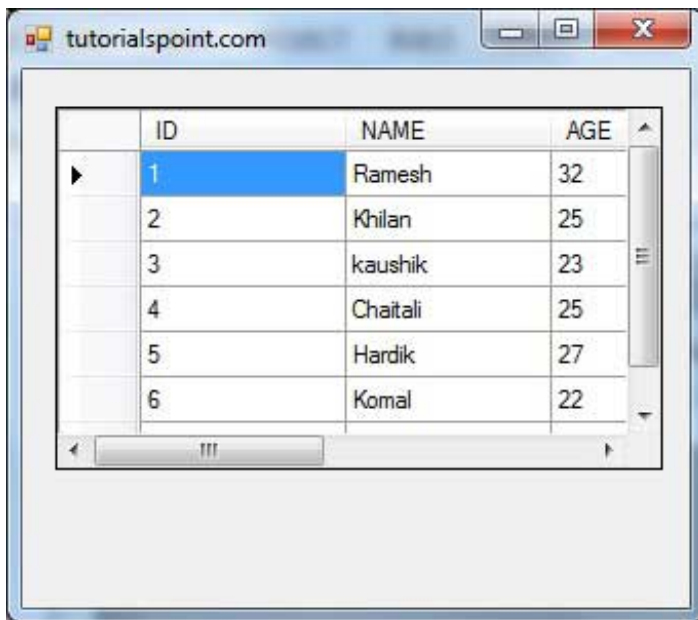11. Save the connection string.

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

13. Select the Preview Data link to see the data in the Results grid:

When the application is run using **Start** button available at the Microsoft Visual Studio tool bar, it will show the following window:
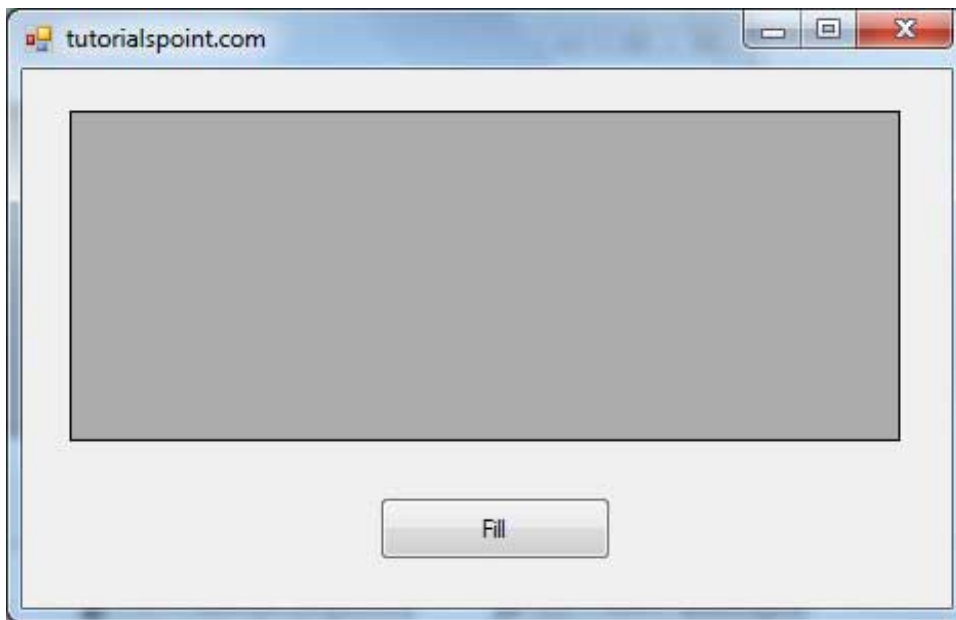
Example 2

In this example, let us access data in a DataGridView control using code. Take the following steps:
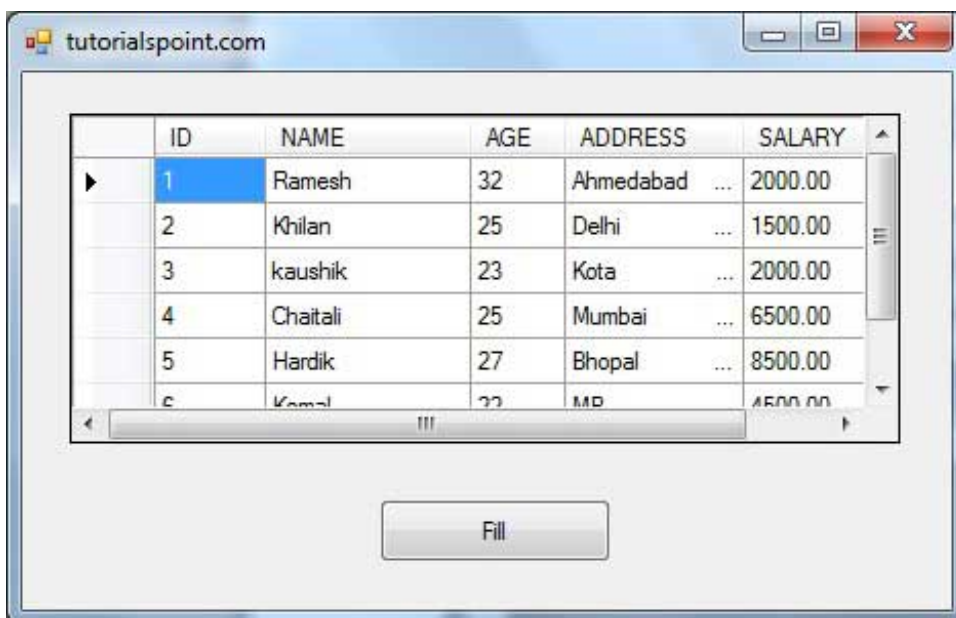
1. Add a DataGridView control and a button in the form.

2. Change the text of the button control to 'Fill'.

3. Double click the button control to add the required code for the Click event of the button, as shown below:

```vb
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:

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.
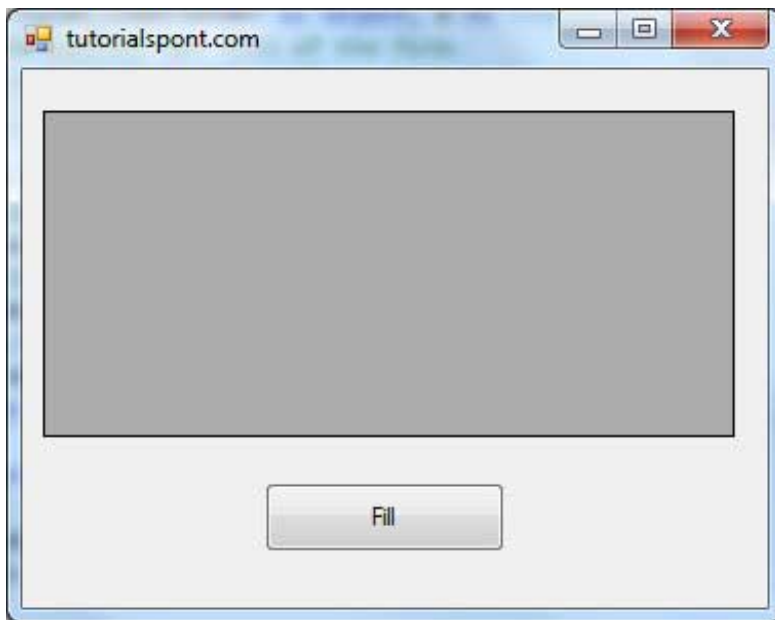
```
Public Class Form1
```

```
    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: