Accessing Databases

Types Available

- Visual Studio 2017 does not ship with SQL Server Express. You can download and install the latest version.
- You can also use an Access database by installing the Access product available from Imagine.
- Visual Studio 2017 includes LocalDB which is included if you select the database tools option when installing or do a modification later. <u>This is the option we will use for this course.</u>

Server Instances

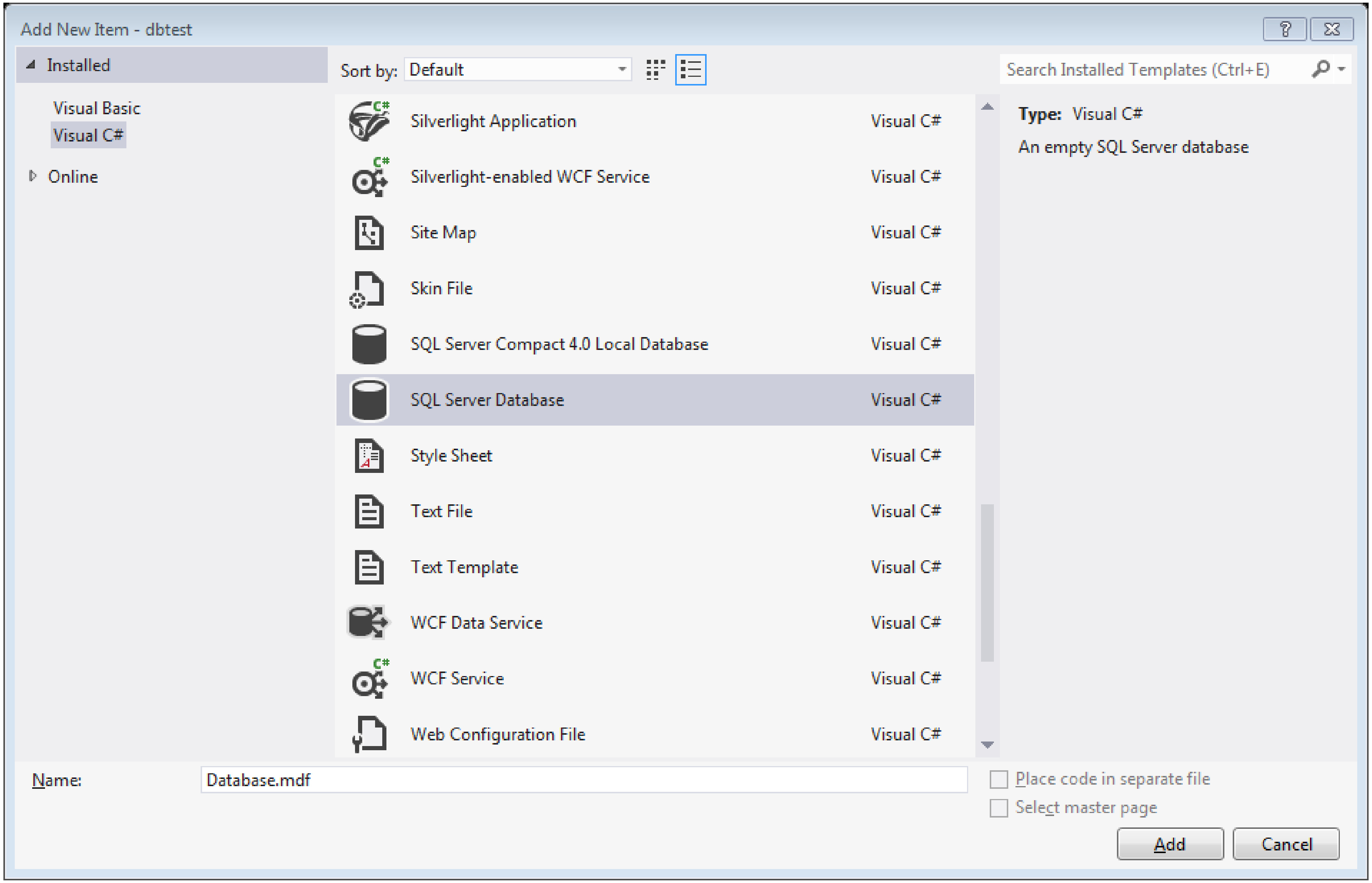
- The main server instance is <sysname>\SQLEXPRESS for SQL Express.
- You can simplify this to .\SQLEXPRESS.
- SQL Express supports a new feature, LocalDB, that makes debugging applications easier.
- I will discuss deployment of the database to the system instance later on.
- The user instance runs automatically if you execute an ASP.NET web page. This is similar to the debugging server.
- Use LocalDB for all assignments. Although installing the full version of SQL Express is of great value, you do not need it to run LocalDB. The data tools option does that.

ADO.NET

- ADO.NET is the .NET library to interface to underlying database providers such as SQL Server, Access, Oracle, or any database product with a suitable provider including legacy standards such as ODBC, OLE DB etc.
- ADO.NET also incorporates the use of XML as a database.
- ADO.NET is based on the earlier ADO product but is significantly different in its use.
- ASP.NET 2 introduced some very powerful controls to vastly simplify the need to use ADO.NET directly, but we will look under the hood first.

Creating a Database in VS2017

1. Add a new database to the App_Data folder by right clicking.
2. Add a table by right clicking Tables in the server explorer and selecting add new table.
3. Change the table n name in the SQL.
4. Enter the columns you want.
5. Click the update tab.
6. Click Update Database in the dialog.



Add New Item - dbtest

- Installed
  - Visual Basic
  - Visual C#
- Online

Sort by: Default

Search Installed Templates (Ctrl+E)

Silverlight Application	Visual C#
Silverlight-enabled WCF Service	Visual C#
Site Map	Visual C#
Skin File	Visual C#
SQL Server Compact 4.0 Local Database	Visual C#
SQL Server Database	Visual C#
Style Sheet	Visual C#
Text File	Visual C#
Text Template	Visual C#
WCF Data Service	Visual C#
WCF Service	Visual C#
Web Configuration File	Visual C#

Type: Visual C#

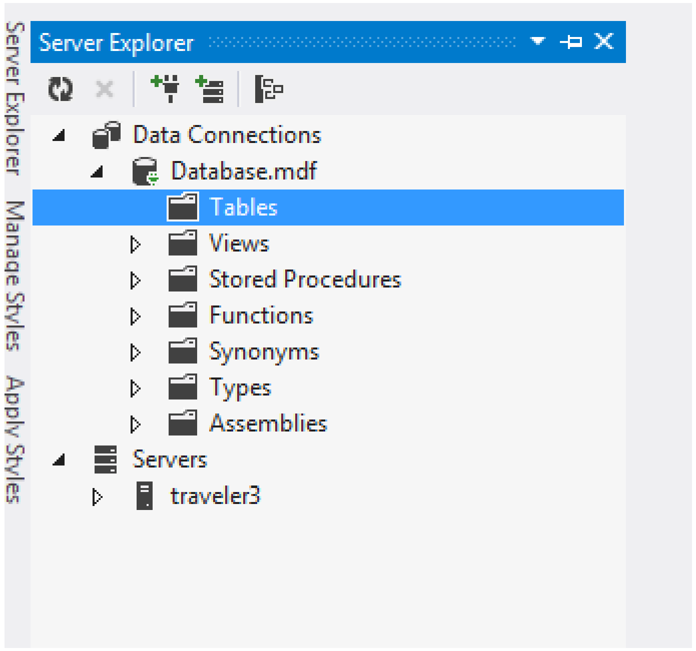
An empty SQL Server database

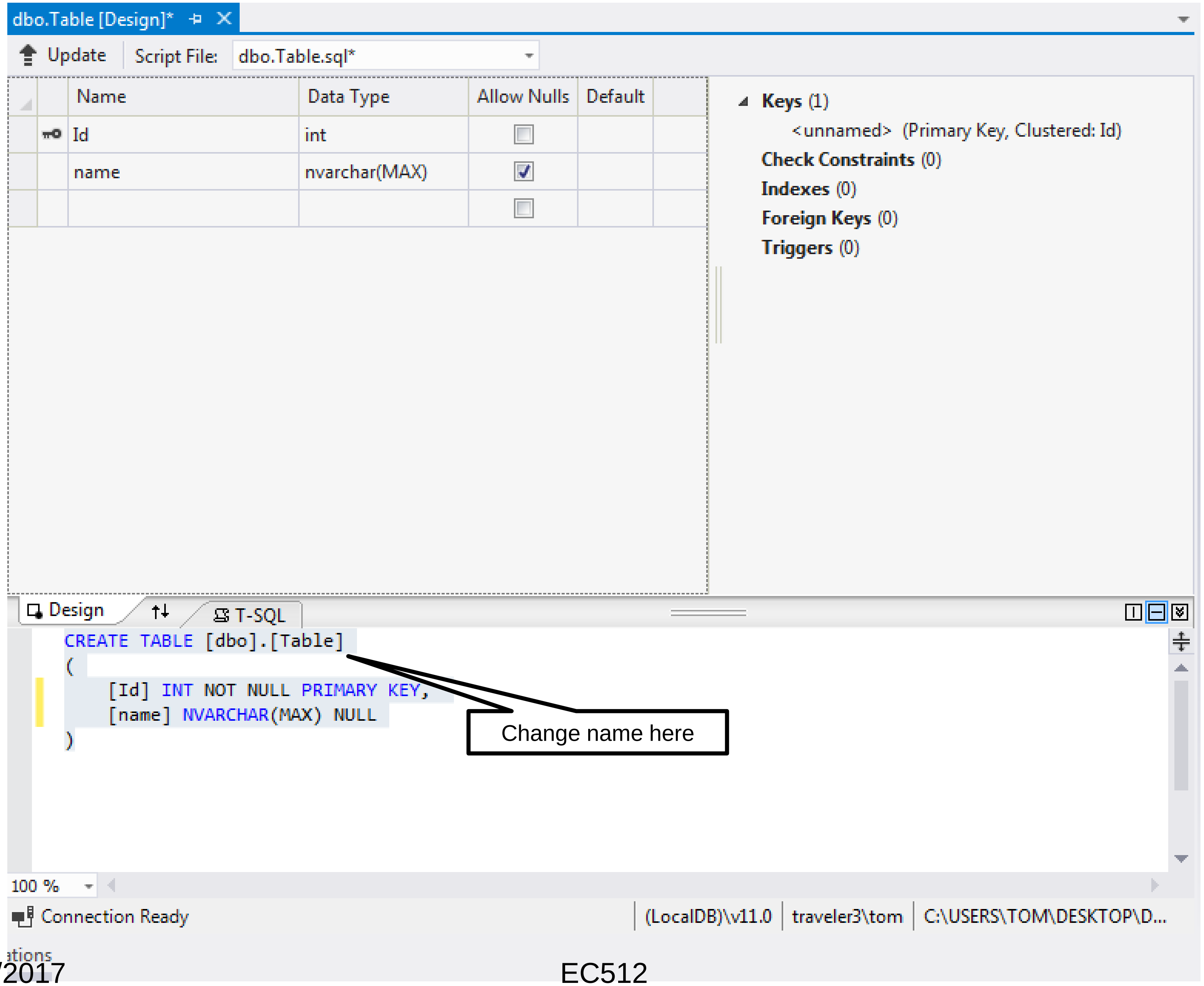
Name: Database.mdf

Place code in separate file

Select master page

Add Cancel


Server Explorer
Manage Styles
Apply Styles
Server Explorer
Data Connections
Database.mdf
Tables
Views
Stored Procedures
Functions
Synonyms
Types
Assemblies
Servers
traveler3



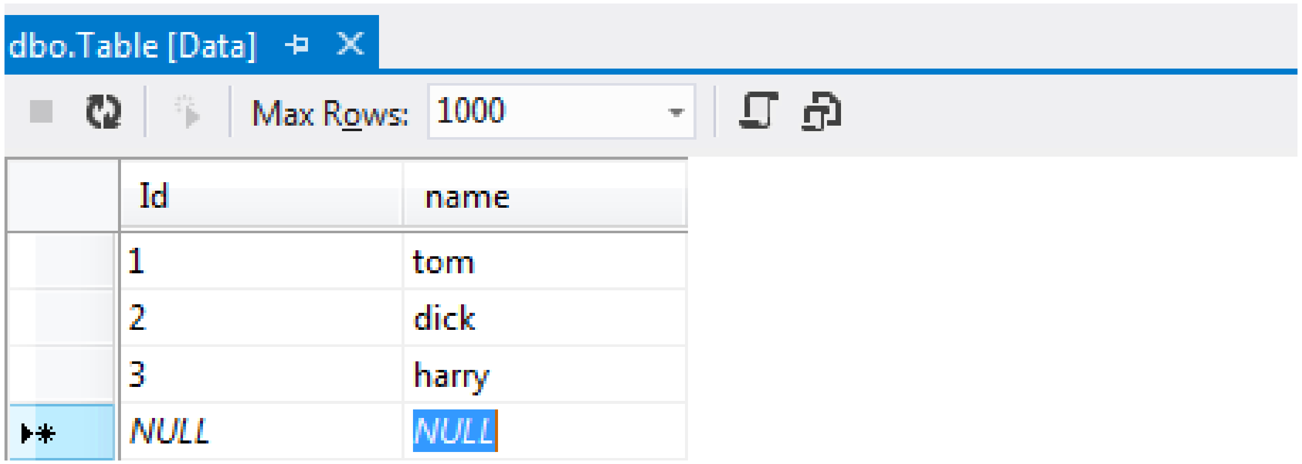
Name	Data Type	Allow Nulls	Default
Id	int	☐	
name	nvarchar(MAX)	☑	
		☐	

Keys (1)
<unnamed> (Primary Key, Clustered: Id)
Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

```
CREATE TABLE [dbo].[Table]
(
    [Id] INT NOT NULL PRIMARY KEY,
    [name] NVARCHAR(MAX) NULL
)
```

Adding Table Data

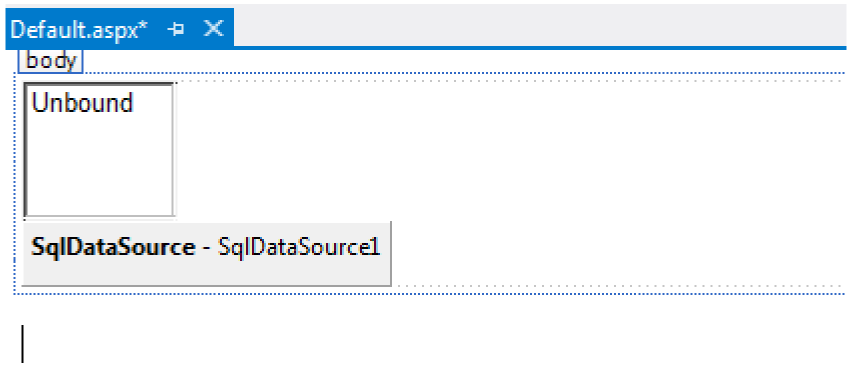
1. Right Data Connections in Server Explorer and click refresh.
2. Right click the table name in the server explorer and select show table data.
3. Add rows of data as needed.
4. Save or close the window.



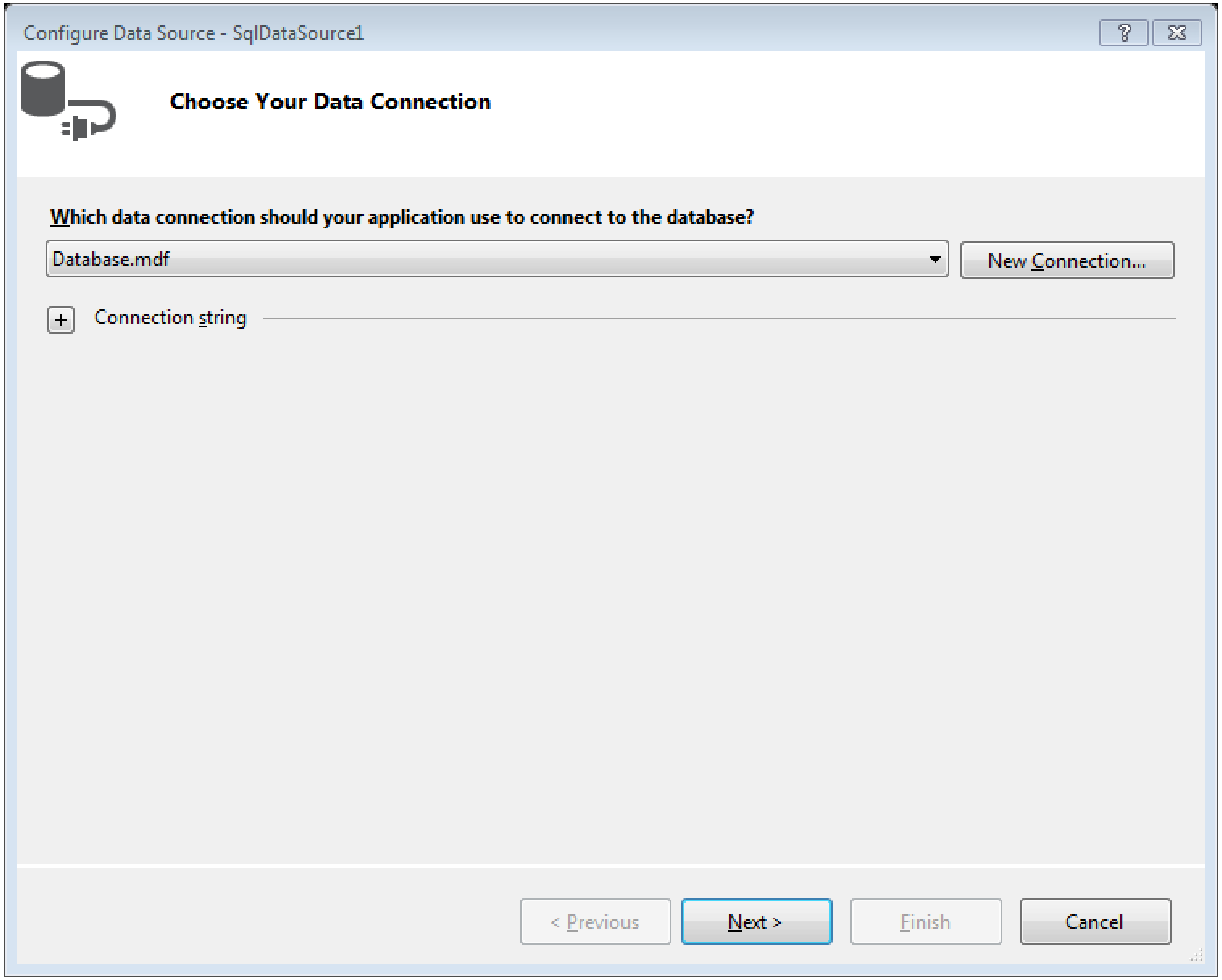
Id	name
1	tom
2	dick
3	harry
NULL	*NULL*

Binding to a Control

- Drag an SQL Data Source control to the page
- Configure the data source to select the database and table.
- Drag a list control to the page.
- Set the data source ID to the data source just created.
- Set data source field to the proper column.
- Bingo! The control is populated by the table in the database.
- Demo



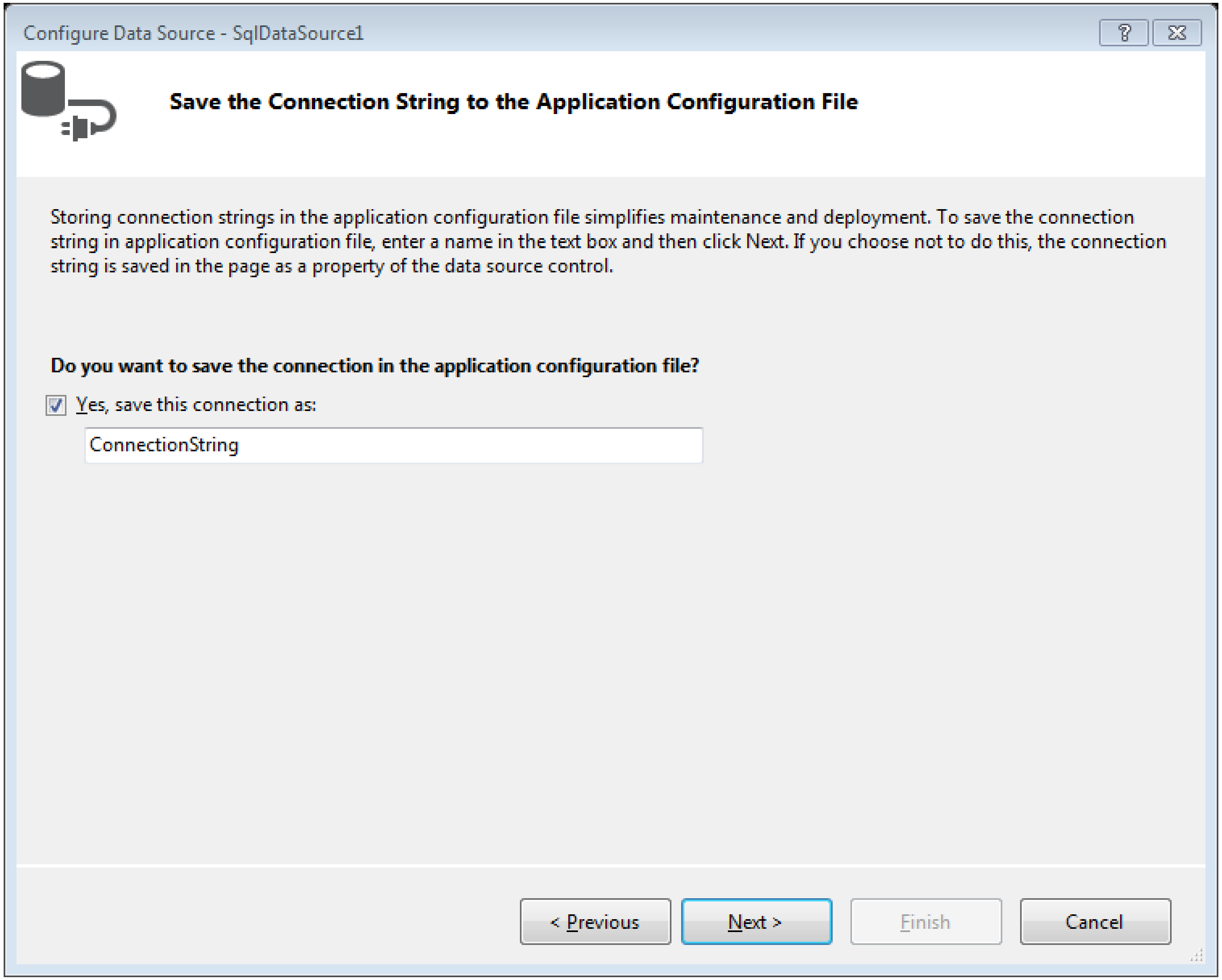
SqlDataSource - SqlDataSource1


Configure Data Source - SqlDataSource1

Choose Your Data Connection

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

Database.mdf



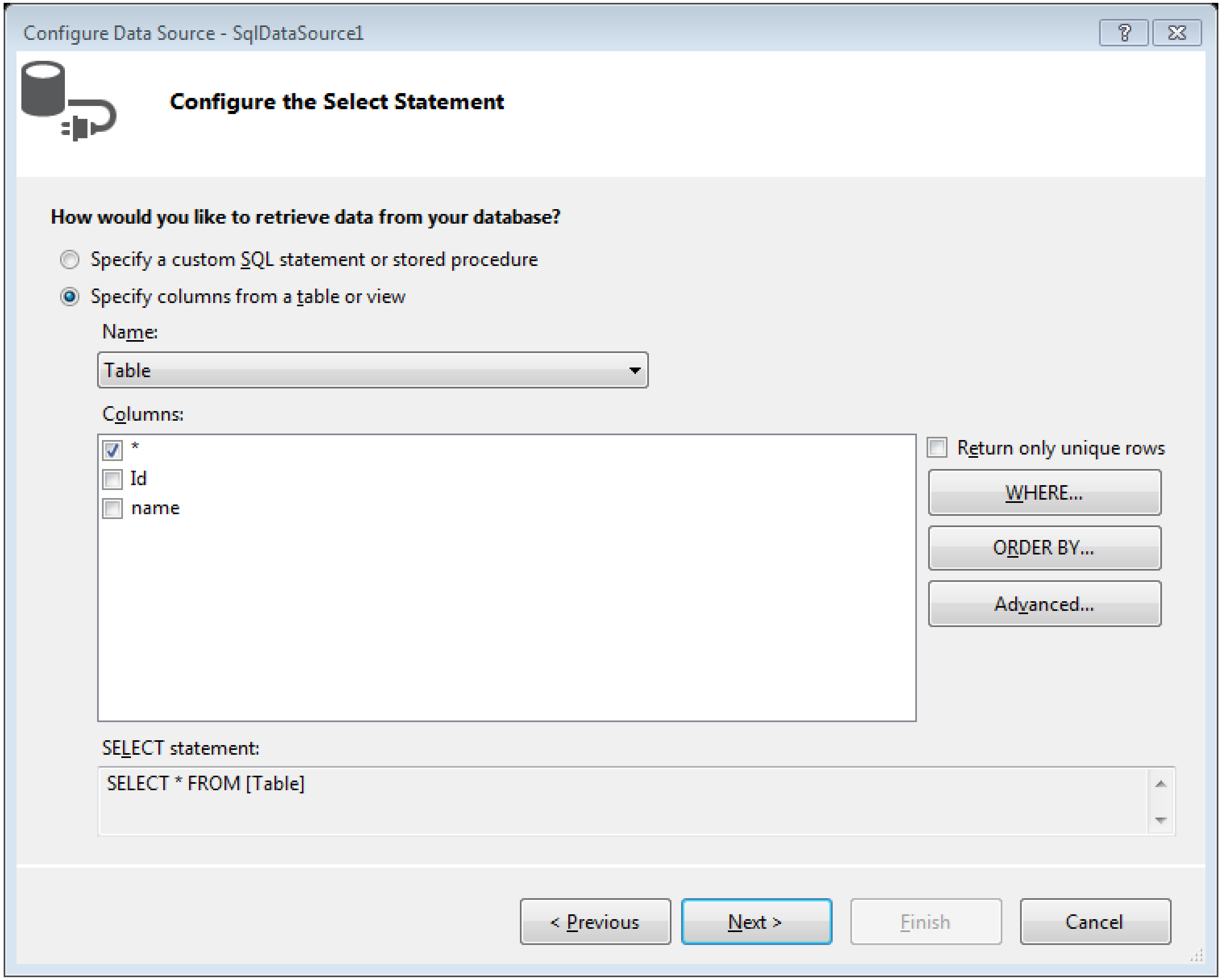
Save the Connection String to the Application Configuration File

Storing connection strings in the application configuration file simplifies maintenance and deployment. To save the connection string in application configuration file, enter a name in the text box and then click Next. If you choose not to do this, the connection string is saved in the page as a property of the data source control.

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

☑ Yes, save this connection as:

ConnectionString


Configure Data Source - SqlDataSource1

Configure the Select Statement

How would you like to retrieve data from your database?

- ○ Specify a custom SQL statement or stored procedure
- ● Specify columns from a table or view

Name: Table

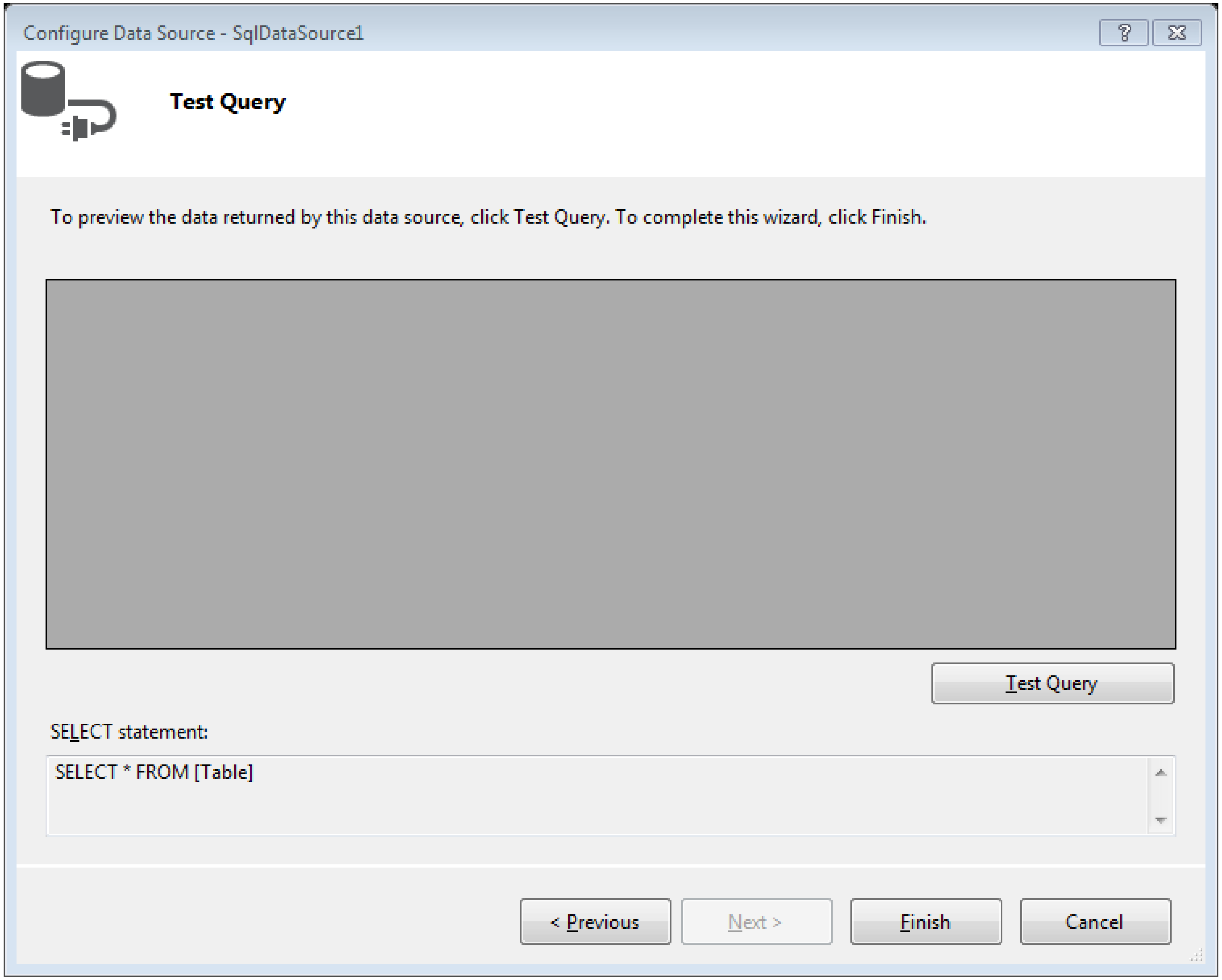
Columns:

- ✓ *
- ☐ Id
- ☐ name

☐ Return only unique rows

SELECT statement:

```
SELECT * FROM [Table]
```



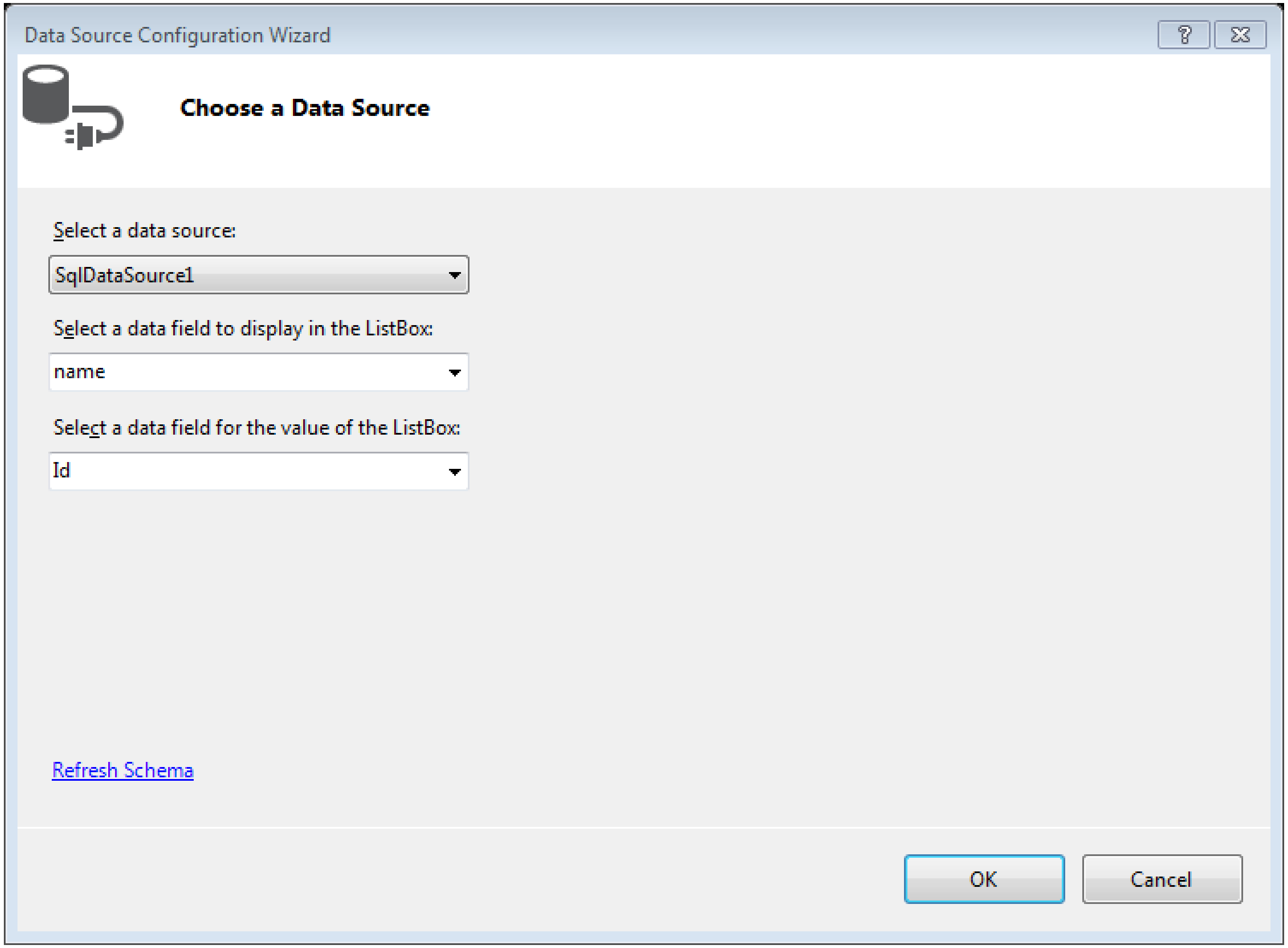
Configure Data Source - SqlDataSource1

Test Query

To preview the data returned by this data source, click Test Query. To complete this wizard, click Finish.

SELECT statement:

```
SELECT * FROM [Table]
```

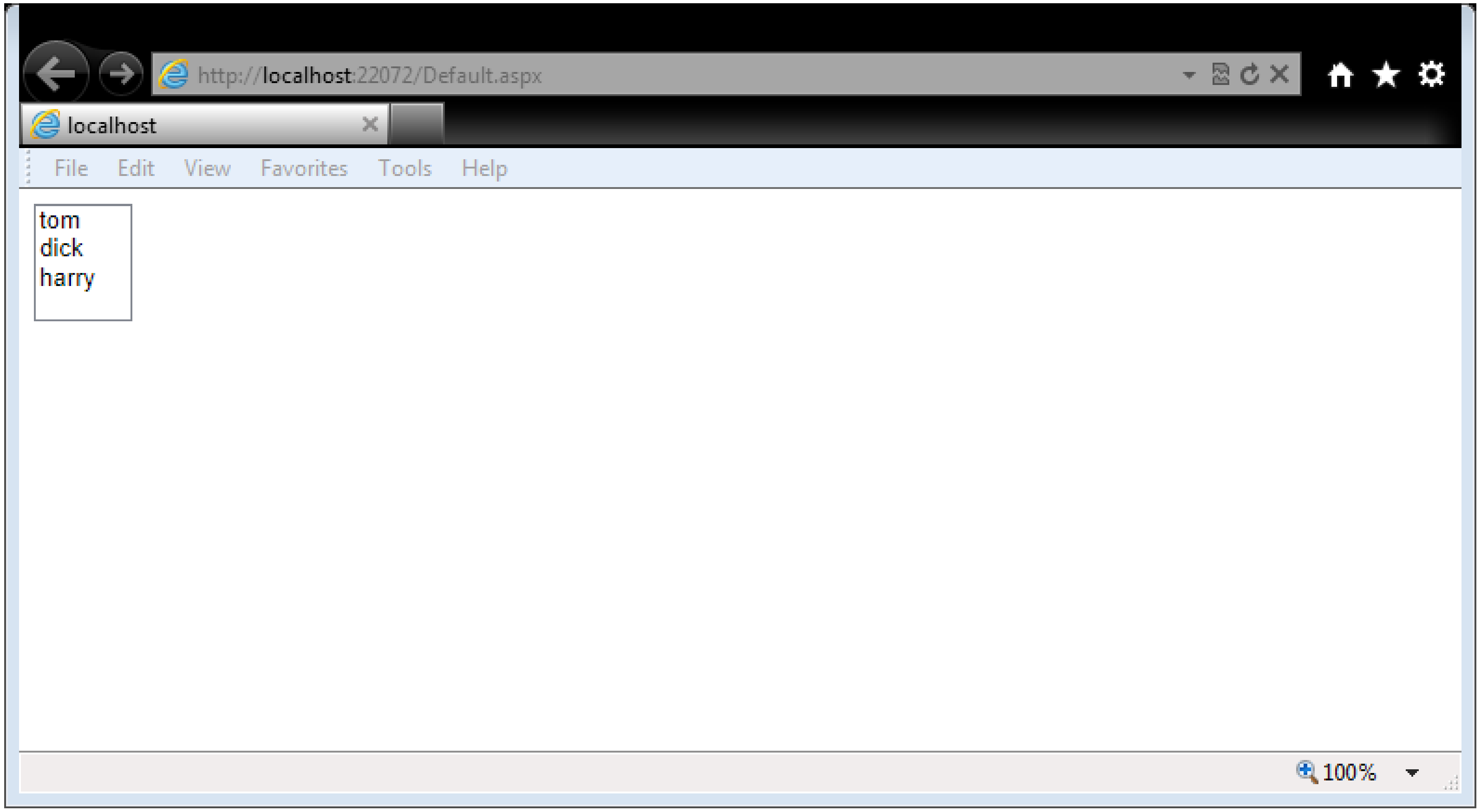

Data Source Configuration Wizard

Choose a Data Source

Select a data source: SqlDataSource1

Select a data field to display in the ListBox: name

Select a data field for the value of the ListBox: Id


http://localhost:22072/Default.aspx
localhost
File Edit View Favorites Tools Help
100%

tom
dick
harry

The Connection String

- The connection string is critical to establishing a connection with the server.
- Here is a connection string for a user instance of SQL Express that attaches the database we just created:
  "Data Source=(LocalDB)\MSSQLLocalDB; AttachDbFilename=|DataDirectory|\Database.mdf;Integrated Security=True"

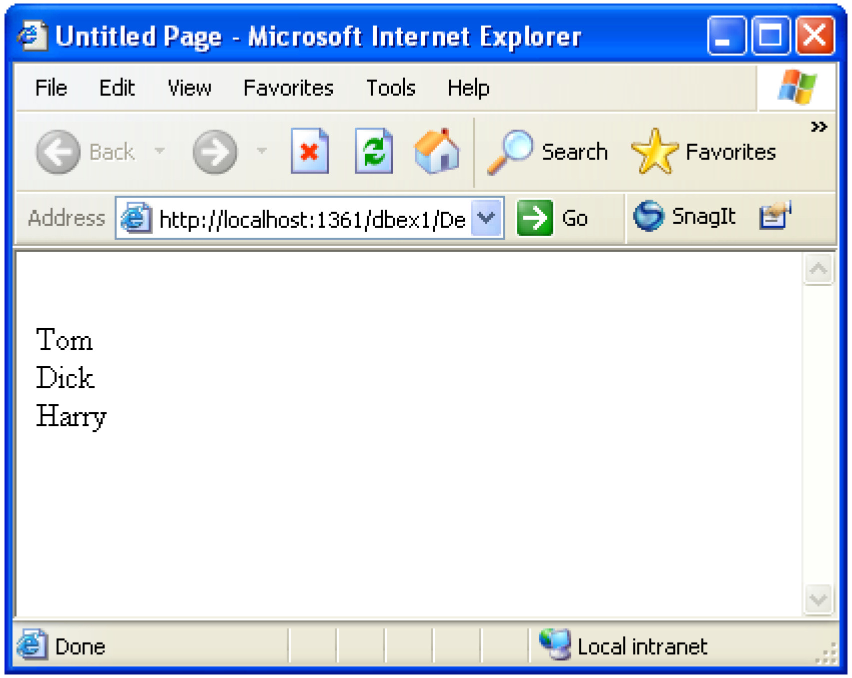
Using a Data Reader

- This is a simple example that you can correlate with the material in the Prosise book.
- The data reader is a sequential and non-updatable access method.
- |DataDirectory| in the connection string indicates the App_Data folder and so there is no need for an absolute pathname.
- Server.MapPath() can also be used for relative paths on the server other than the data folder. (Not a good idea.)

Using a Data Reader (Ado1)

```
using System.Data.SqlClient;
public partial class _Default : System.Web.UI.Page
{
    private const string connString = @"Data Source =
       LocalDB)\v11.0;AttachDbFilename=|DataDirectory|\MyData.mdf;Integrated ….
    protected void Page_Load(object sender, EventArgs e)
    {
        SqlConnection conn = new SqlConnection(connString);
        conn.Open();
        SqlCommand cmd = new SqlCommand("SELECT * FROM Table1", conn);
        SqlDataReader rdr = cmd.ExecuteReader();
        while (rdr.Read())
        {
            string s = (string)rdr["name"];
            Response.Write("<br />" + s);
        }
        conn.Close();
    }
}
```

SQL



```
Tom
Dick
Harry
```

SQL

- Structured Query Language is the language of virtually all database engines.
- There are many books and web sites with tutorials.
- Do a Google search to find one you like.
- MS SQL Server’s version is called Transact SQL.

Non-query Commands

```
conn.Open ();
SqlCommand cmd = new SqlCommand ();
cmd.CommandText = "delete from titles where title_id =
'BU1032'";
cmd.Connection = conn;
cmd.ExecuteNonQuery (); // Execute the command
conn.Close();
```

Insertions

```
 conn.Open ();
   SqlCommand cmd = new SqlCommand
       ("insert into titles (title_id, title, type, pubdate) " +
       "values ('JP1001', 'Programming Microsoft .NET', " +
       "'business', 'May 2002')", conn);
   cmd.ExecuteNonQuery ();
conn.Close();
```

Updates

```
 conn.Open ();
    SqlCommand cmd = new SqlCommand
        ("update titles set title_id = 'JP2002' " +
        "where title_id = 'JP1001'", conn);
    cmd.ExecuteNonQuery ();
conn.Close ();
```

Deletes

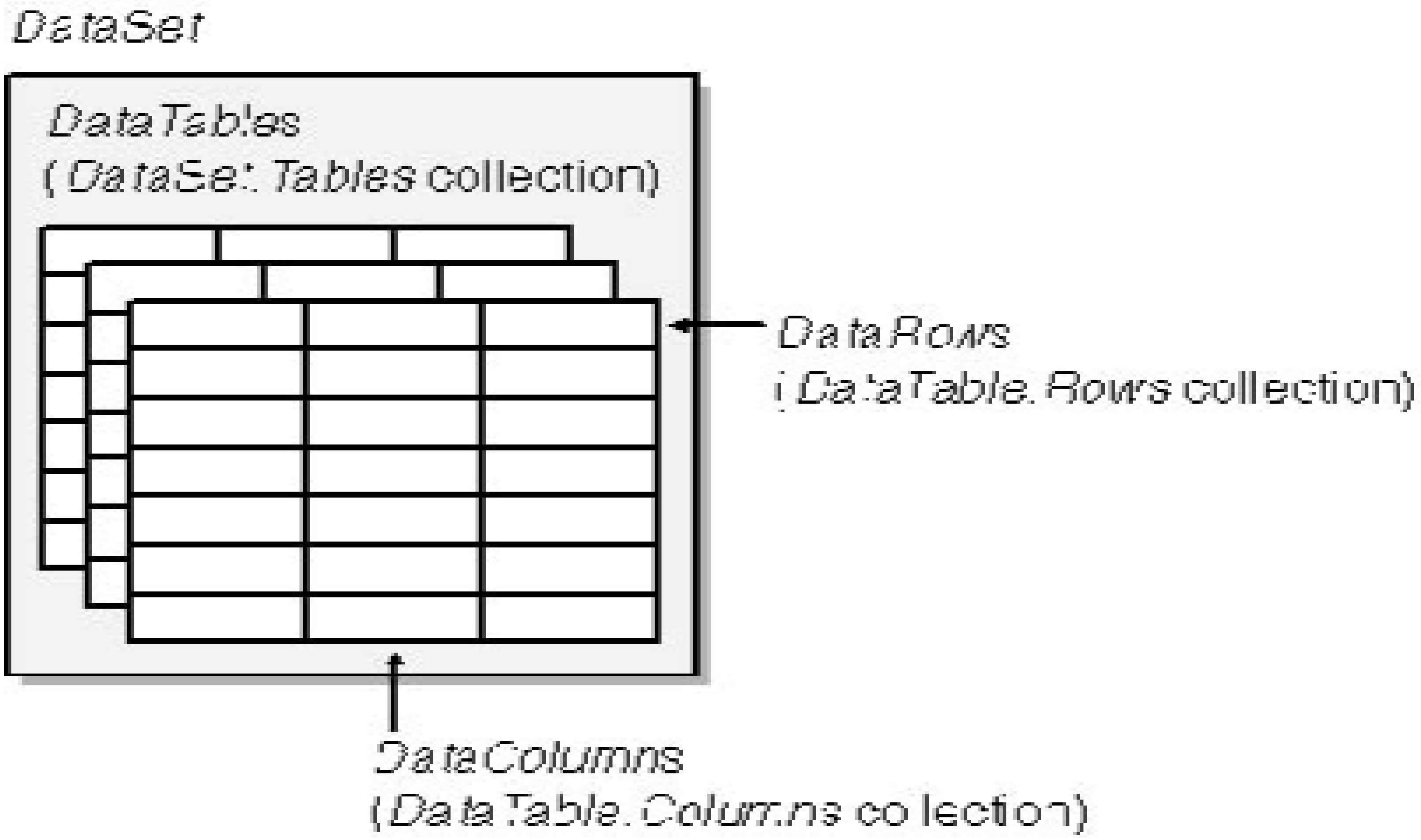
```
 conn.Open ();
    SqlCommand cmd = new SqlCommand
        ("delete from titles where title_id =
   'JP2002'", conn);
    cmd.ExecuteNonQuery ();
Conn.Close();
```

The *ExecuteScalar* Method

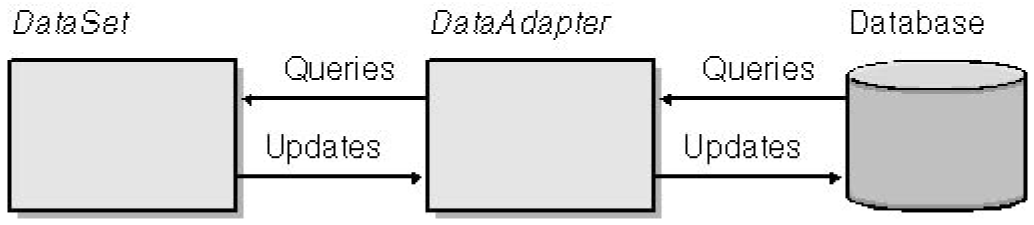
- Used to obtain aggregate information such as counts etc.
- See the Prosise book.

DataSets and DataAdapters

- Used to obtain and manipulate a set of data rows.
- The set can be updated back on the server.


DataSet
DataTables
(DataSet.Tables collection)
DataRows
(DataTable.Rows collection)
DataColumns
(DataTable.Columns collection)

The DataAdapter



Filling a DataAdapter

```
SqlDataAdapter adapter = new SqlDataAdapter ("select *
    from titles", connString);
    DataSet ds = new DataSet ();
adapter.Fill (ds, "Titles");
```

Inserting Rows

```
SqlDataAdapter adapter = new SqlDataAdapter ("select * from titles",
   connString");

DataSet ds = new DataSet ();
adapter.Fill (ds, "Titles");

// Create a new DataRow
DataTable table = ds.Tables["Titles"];
DataRow row = table.NewRow ();

// Initialize the DataRow
row["title_id"] = "JP1001";
row["title"] = "Programming Microsoft .NET";
row["price"] = "59.99";
row["ytd_sales"] = "1000000";
row["type"] = "business";
row["pubdate"] = "May 2002";

// Add the DataRow to the DataTable
table.Rows.Add (row);
```

Updating to the Server

```
// Update the database
adapter.Update (table);
```