Oracle® Rdb Developer Tools for Visual Studio

Developer's Guide
Release 7.3.3.0
May 2014

Oracle Rdb Data Provider for .NET Developer's Guide, Release 7.3.3.0

Primary Author: Jim Murray.

Contributing Author:

Contributor:

Contents

Send Us Your Comments

Oracle Rdb Developer Tools for Visual Studio Developer's Guide, Release 7.3.3.0

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: nedc-doc_us@oracle.com
- FAX — 603-897-3825 Attn: Oracle Rdb
- Postal service:
  Oracle Corporation
  Oracle Rdb Documentation
  One Oracle Drive
  Nashua, NH 03062-2804
  USA

If you would like a reply, please give your name, address, telephone number, and electronic mail address (optional).

If you have problems with the software, please contact your local Oracle Support Services.

Preface

This document is your primary source of introductory and usage information for Oracle Rdb Developer Tools for Visual Studio.

This preface contains these topics:

- Audience
- Access to Oracle Support
- Organization
- Related Documentation
- Conventions

Audience

Oracle Rdb Developer Tools for Visual Studio Developer's Guide is intended for developers who are developing applications to access an Oracle Rdb database using Oracle Rdb Data Provider for .NET. This documentation is also valuable to systems analysts, project managers, and others interested in the development of database applications.

To use this document, you must be familiar with Microsoft .NET Framework classes and ADO.NET and have a working knowledge of application programming using Microsoft C#, Visual Basic, or C++.

Users should also be familiar with the use of Structured Query Language (SQL) to access information in relational database systems.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Organization

This document contains:

- **Chapter 1, Introducing Oracle Rdb Developer Tools for Visual Studio.**
  Provides an overview of Oracle Rdb Developer Tools for Visual Studio.
- **Chapter 2, Installing Oracle Rdb .NET Products**
  Describes how to install Oracle Rdb Data Provider for .NET and provides system requirements. Read this chapter *before* installing or using Oracle Rdb Developer Tools for Visual Studio.
- **Chapter 3, Using Oracle Rdb Developer Tools for Visual Studio**
  Describes how to create a database connection using ORDP.NET within Visual Studio.
- **Chapter 4, Building a Simple .NET Application Using ORDP.NET**
  Provides an example on how you may build a simple application with database access using ORDP.NET.
- **Chapter 5, Retrieving and Updating with ORDP.NET**
  Provides an example of data retrieval and update using the features of Visual Studio and data access using ORDP.NET.
- **Chapter 6, ORDP.NET and Visual Studio Wizards**
  Provides an example on using standard Visual Studio wizards to automatically generate the code to access data from your Rdb database.
- **Chapter 7, ORDP.NET and drag and drop**
  Provides an example of using the drag and drop features of Visual Studio to generate code for your application.
- **Chapter 8, Configuration wizard and generating queries**
  Provides an example of using Table Adapter Query Configuration Wizard to generate queries on your database.
- **Chapter 9, Copying a Form**
  Describes how to copy a form as required when following the examples provided in this document.
- **Chapter 10. Oracle Rdb Entity Framework Provider**
  Describes how to use ORDP.NET within the Entity Framework using standard Entity Framework tools and features of Visual Studio.
- **Chapter 11. Unsupported features**
  Specifies what Visual Studio features are currently not supported by ORDT.
- **Glossary**
  Defines terms used in this document.

Related Documentation

For more information, see these Oracle Rdb resources:

- *Oracle Rdb7 Guide to Database Design and Definition*
- *Oracle Rdb7 Guide to Database Performance and Tuning*
- *Oracle Rdb Introduction to SQL*
- *Oracle Rdb 7.2 SQL Reference Manual*
- *Oracle Rdb Guide to SQL Programming*
- *Oracle SQL/Services Server Configuration Guide*
- *Guide to Using the Oracle Rdb Oracle SQL/Services (tm) Client API*

- *Oracle Rdb JDBC Driver Users Guide*

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Rdb web site:
http://www.oracle.com/technetwork/database/rdb

For additional information, see:
http://msdn.microsoft.com/netframework

Conventions

Oracle Rdb Data Provider for .NET is often referred to as ORDP.NET.

Oracle Rdb is often referred to as Rdb.

Hewlett-Packard Company is often referred to as HP.

The following conventions are used in this document:

word	A lowercase word in a format example indicates a syntax element that you supply.
[ ]	Brackets enclose optional clauses from which you can choose one or none.
{ }	Braces enclose clauses from which you must choose one alternative.
...	A horizontal ellipsis means you can repeat the previous item
. . .	A vertical ellipsis in an example means that information not directly related to the example has been omitted.

Conventions in Code Examples

Code examples illustrate SQL or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT last_name FROM employees WHERE last_name = 'TOLIVER';
```

Example Screen Shots

Unless otherwise stated, examples and screen shots found within this document relate to Visual Studio 2012.

As the user interface for Visual Studio may change between versions, if you are using another version of Visual Studio, please consult your Visual Studio documentation to determine how to carry out the same operations described in the provided example.

Show Navigation

Chapter 1
Introducing Oracle Rdb Developer Tools for Visual Studio.

Oracle Rdb Developer Tools for Visual Studio (ORDT) is a set of application tools that integrate with the Visual Studio environment. These tools provide graphical user interface access to Oracle Rdb functionality, enable the user to perform a wide range of application development tasks, and improve development productivity and ease of use.

ORDT includes:

- Oracle Rdb Data Provider for .NET (ORDP.NET).

  A standard .NET Data Provider that allows access to Oracle Rdb databases from the .NET environment

- Oracle Rdb DDEX Provider.

  A standard .NET DDEX provider that integrates ORDP.NET functionality into Visual Studio.

- Oracle Rdb Entity Framework Provider.

  A standard .NET Entity Framework provider that integrates ORDP.NET functionality into Entity Framework.

Chapter 2   Installing Oracle Rdb .NET Products

See the Oracle Rdb Developer Tools for Visual Studio Release Notes for information on how to install Oracle Rdb .NET products.

Show Navigation

Chapter 3 Using Oracle Rdb Developer Tools for Visual Studio

This chapter contains:

3.1 Using Oracle Rdb Developer Tools

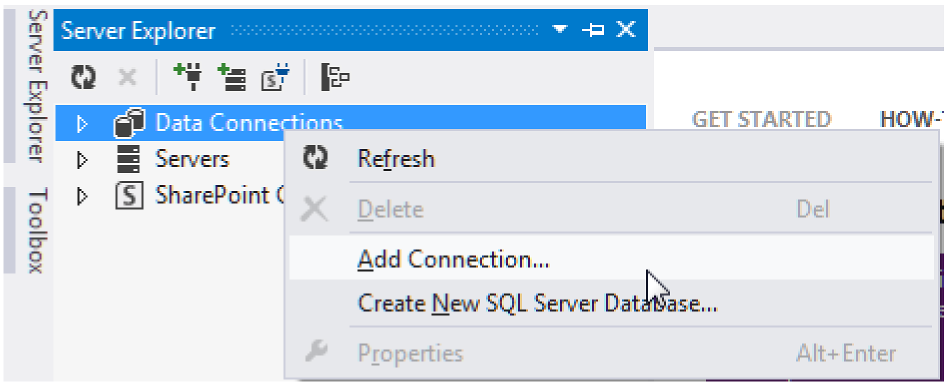
Oracle Rdb Developer Tools for Visual Studio (ORDT) is a tightly integrated Add-in for Visual Studio. Using enhancements that ORDT brings to the Server Explorer, you can automatically create tables, indexes, constraints, data connections and other database schema objects. Additionally you can automatically generate application code.

3.2 Connecting to the Oracle Rdb Database

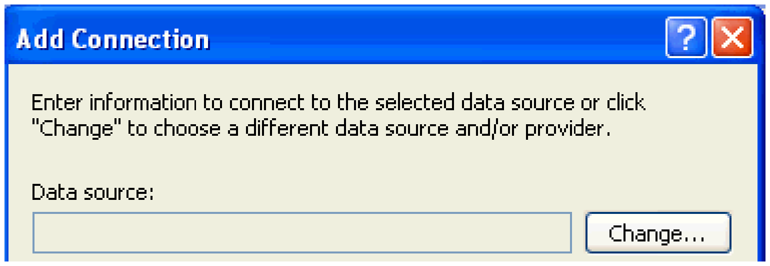
This section shows you how to use the Server Explorer to connect to the Oracle Rdb Database for the purpose of automatically creating or modifying database schema objects.

To connect to the database:

Step1: From the View menu, select **Server Explorer**.
Step2: In Server Explorer, right-click **Data Connections**.
Step3: Select **Add Connection**.



Step4: If you have selected any Data Source in the past, the **Add Connection** window appears, determine if the Data source says Oracle Rdb Database (RdbClient). If it does, skip to Step 6.



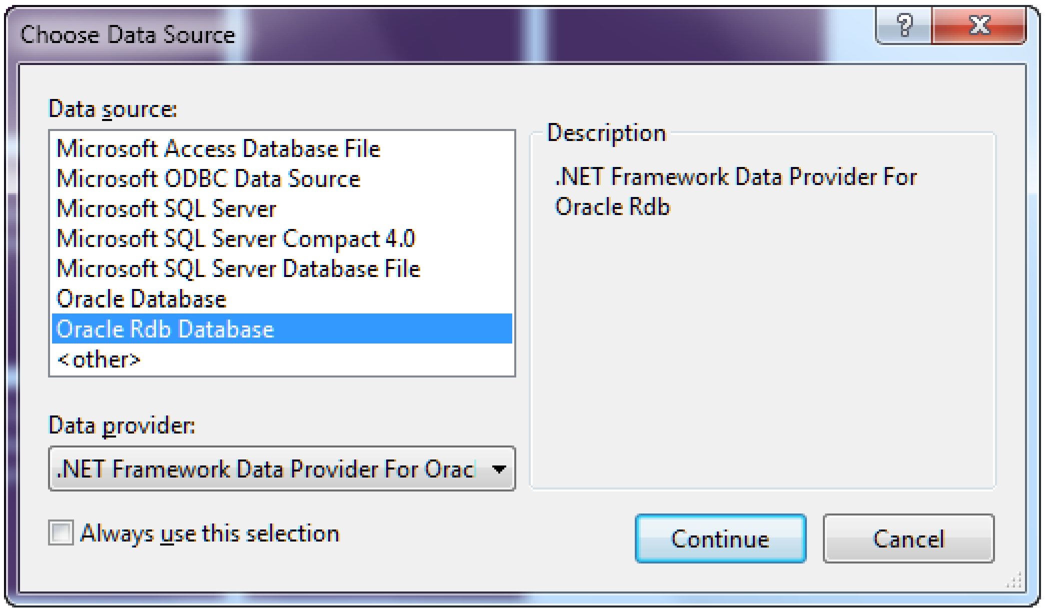
Add Connection

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

Data source:

If Data source does not say Oracle Rdb Database (RdbClient), select **Change**.

If you have not already selected a Data source previously, or you have selected **Change**, the Choose Data Source window will appear.



Choose Data Source

Data source:

- Microsoft Access Database File
- Microsoft ODBC Data Source
- Microsoft SQL Server
- Microsoft SQL Server Compact 4.0
- Microsoft SQL Server Database File
- Oracle Database
- Oracle Rdb Database
- <other>

Description

.NET Framework Data Provider For Oracle Rdb

Data provider: .NET Framework Data Provider For Orac

Always use this selection

Step5: Choose Oracle Rdb Database and then select .NET Framework Data Provider For Oracle Rdb.

Step6: On the Connection Details tab, in the Add Connection window, enter the following information:

To connect to a database using a SQL/Services universal service:

For **Server**, enter the node address and a valid SQL/Services service that will be running on the selected node. Remember to separate the node address and service name with a colon (:). In this example we assume that a universal service called GENERIC is running on the selected node.

The node address must be valid TCP/IP node specification.

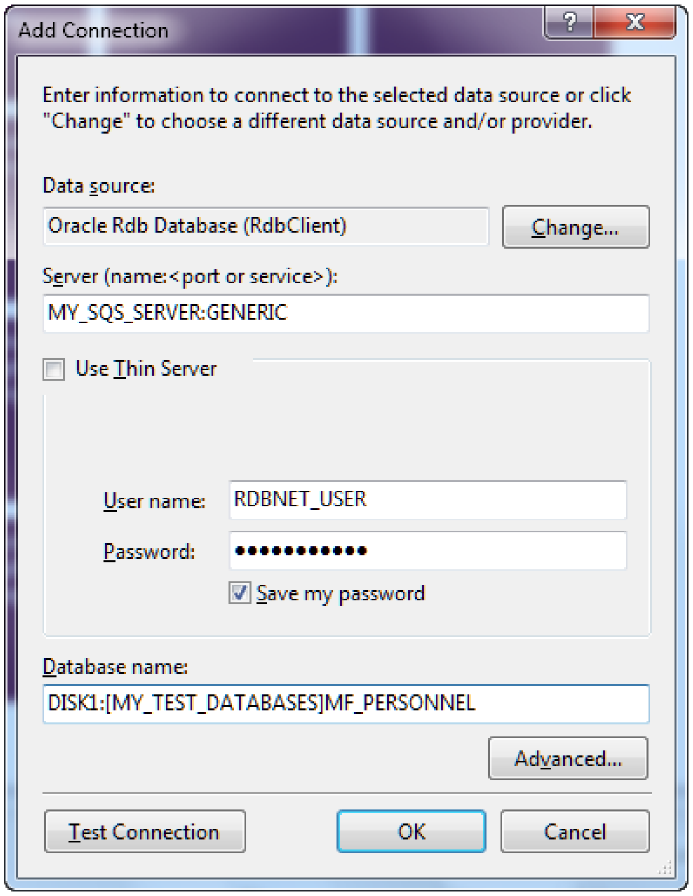
For **User name**, enter a valid username for the database you will select.

For **Password**, enter the password for that user

To save the password for future sessions, check the **Save password** box.

For **Database name**, Enter a valid database file specification. Depending on the setup of SQL/Services on the server node, this database specification may include device and directory information as well as the database filename and may use OpenVMS logical names.

The **Connection name** should be generated automatically from the **Database name**.


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: Oracle Rdb Database (RdbClient)

Server (name:<port or service>): MY_SQS_SERVER:GENERIC

☐ Use Thin Server

User name: RDBNET_USER

Password: ●●●●●●●●●●●

☑ Save my password

Database name: DISK1:[MY_TEST_DATABASES]MF_PERSONNEL

To connect to a database using a SQL/Services database service:

For **Server**, enter the node address and a valid SQL/Services database service that will be running on the selected node. Remember to separate the node address and service name with a colon `(:)`.

In this example we assume that a database service called MF_PERS is running on the selected node. The node address must be valid TCP/IP node specification.

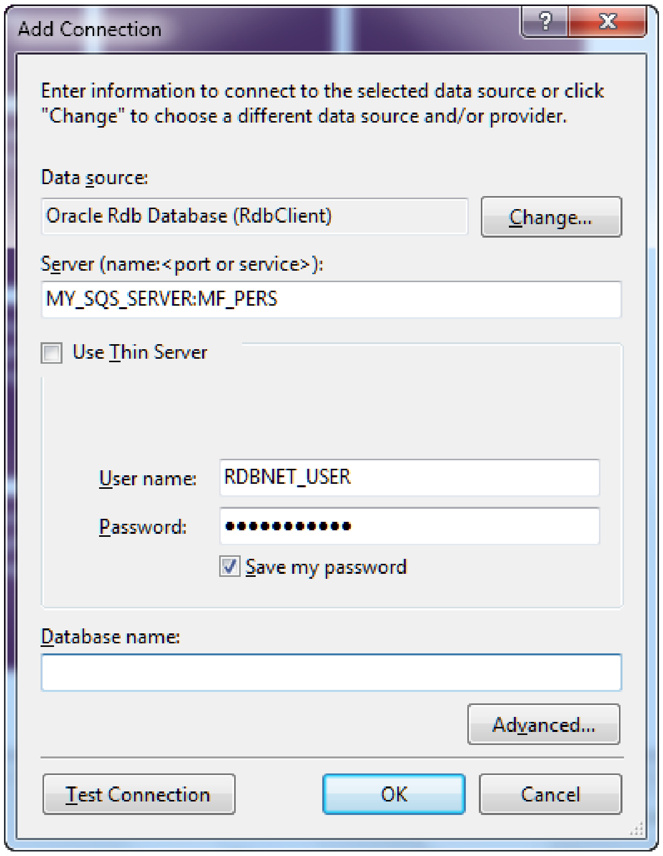
For **User name**, enter a valid username for the database you will select.

For **Password**, enter the password for that user

To save the password for future sessions, check the **Save** password box.

Leave the **Database name** blank as this is not required when a database service is used.

The Connection name should be generated automatically from the Database name.


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: Oracle Rdb Database (RdbClient)

Server (name:<port or service>): MY_SQS_SERVER:MF_PERS

[ ] Use Thin Server

User name: RDBNET_USER

Password: ●●●●●●●●●●●

[✓] Save my password

Database name:

To connect to a database using a JDBC Thin Server:

For **Server**, enter the node address and port for a valid JDBC server that will be running on the selected node. Remember to separate the node address and port with a colon `(:)`. The node address must be valid TCP/IP node specification.

Click the **Use Thin Server** box.

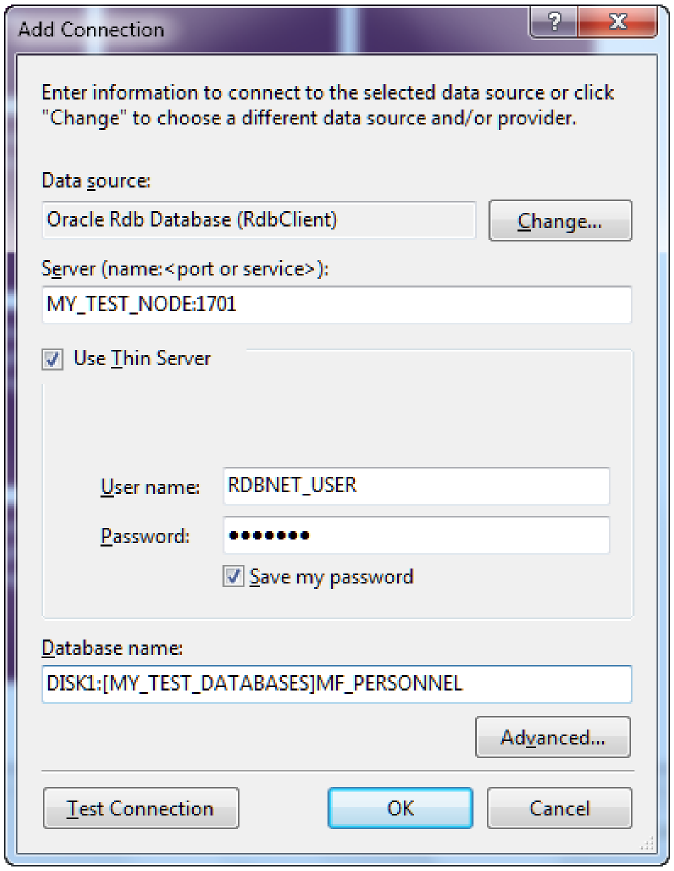
For **User name**, enter a valid username for the database you will select.

For **Password**, enter the password for that user

To save the password for future sessions, check the **Save password** box.

For **Database name**, Enter a valid database file specification. This database specification may include device and directory information as well as the database filename and may use OpenVMS logical names.

The **Connection name** should be generated automatically from the **Database name**.



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

Data source: Oracle Rdb Database (RdbClient)

Server (name:<port or service>): MY_TEST_NODE:1701

☑ Use Thin Server

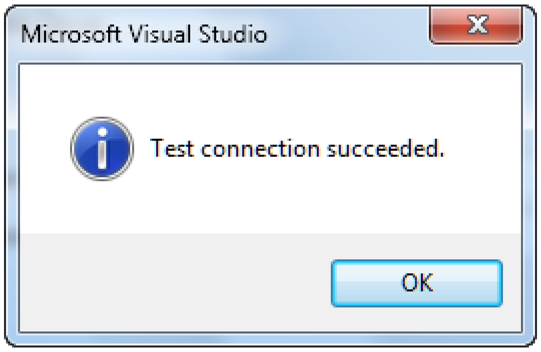
User name: RDBNET_USER

Password: •••••••

☑ Save my password

Database name: DISK1:[MY_TEST_DATABASES]MF_PERSONNEL

Step7: Click Test connection.


Microsoft Visual Studio

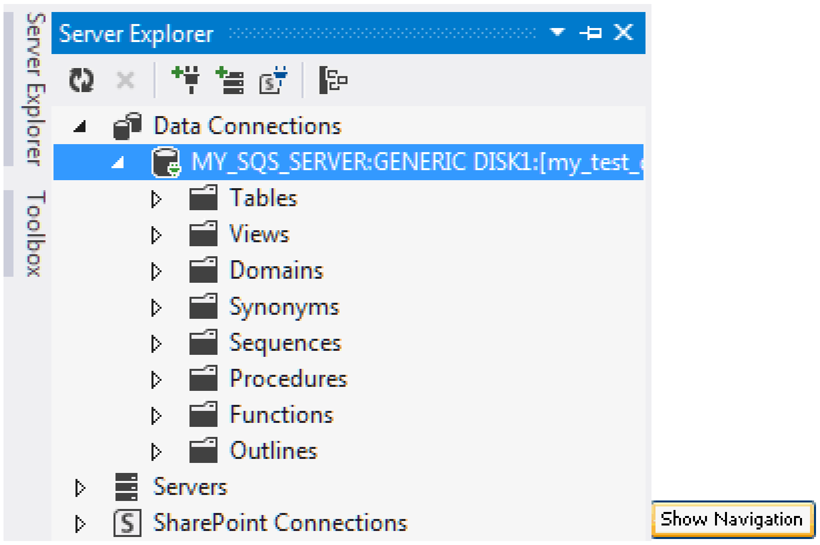
Test connection succeeded.

The test should succeed. Click OK.

If the test fails, it may be due to one or more of the following issues that you must address before proceeding with further steps:

- The database listener or server is not started.
- The database connectivity is not properly configured.
- You do not have the correct user name, or password.

Step8: In the Add Connection window, click OK.
Step9: In the Server Explorer, expand the database connection to show the contents of the `database` . You should see Tables, Views, Procedures, Functions, Packages, Synonyms, Sequences, and so on.



Chapter 4 Building a Simple .NET Application Using ORDP.NET

This chapter contains:

4.1 Creating a New Project

Visual Studio groups all development code that you create into containers known as projects. Simpler projects often contain only one file. In this section, you will learn how to create a new development project.

The application you build in this chapter serves as a starting point for work in subsequent chapters, so it is important to follow the order of this guide.

Note:

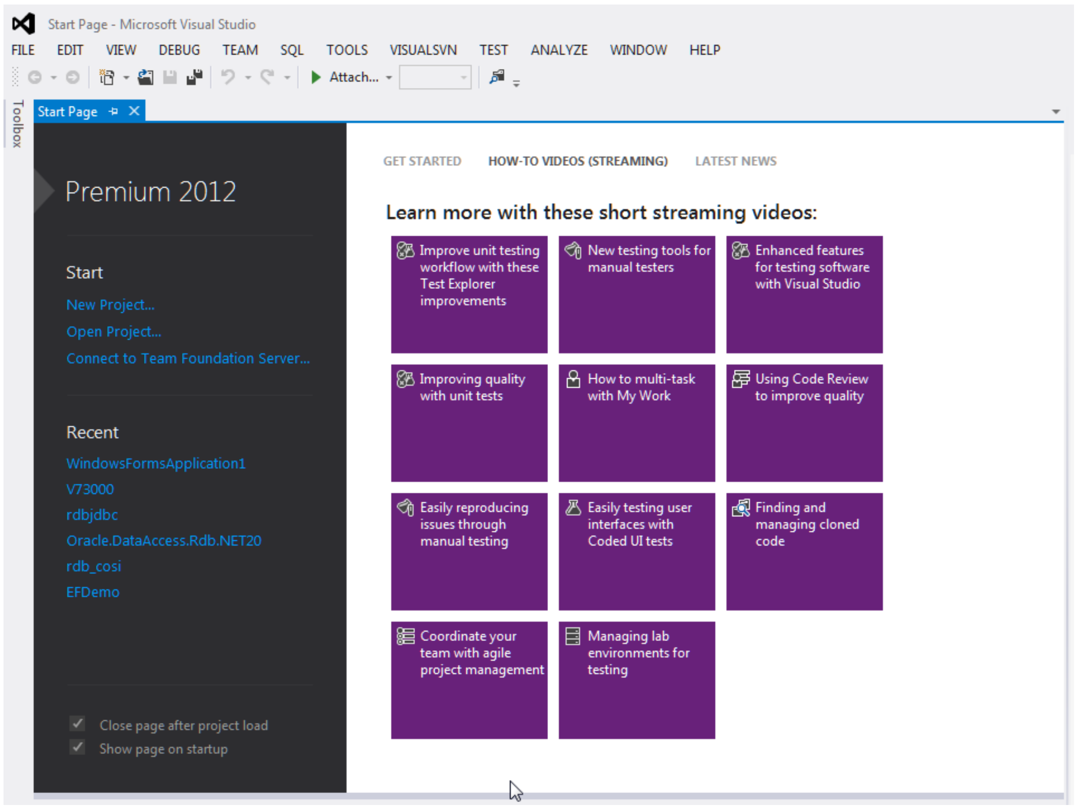
When necessary, instructions specify **Visual C#** or **Visual Basic.**

To start a new project:

Step1: Start Visual Studio.

Open the **Start menu**, select **All Programs**, and then select **Microsoft Visual Studio 2012.**

The Microsoft Visual Studio IDE environment appears.


Start Page - Microsoft Visual Studio
FILE EDIT VIEW DEBUG TEAM SQL TOOLS VISUALSVN TEST ANALYZE WINDOW HELP
Attach...
Toolbox
Start Page
Premium 2012
Start
New Project...
Open Project...
Connect to Team Foundation Server...
Recent
WindowsFormsApplication1
V73000
rdbjdbc
Oracle.DataAccess.Rdb.NET20
rdb_cosi
EFDemo
Close page after project load
Show page on startup
GET STARTED
HOW-TO VIDEOS (STREAMING)
LATEST NEWS
Learn more with these short streaming videos:
Improve unit testing workflow with these Test Explorer improvements
New testing tools for manual testers
Enhanced features for testing software with Visual Studio
Improving quality with unit tests
How to multi-task with My Work
Using Code Review to improve quality
Easily reproducing issues through manual testing
Easily testing user interfaces with Coded UI tests
Finding and managing cloned code
Coordinate your team with agile project management
Managing lab environments for testing

Step2: In the Start Page, under the Start heading, click **New Project...**.

Alternatively, from the **File** menu, select **New**, and then select **Project**.

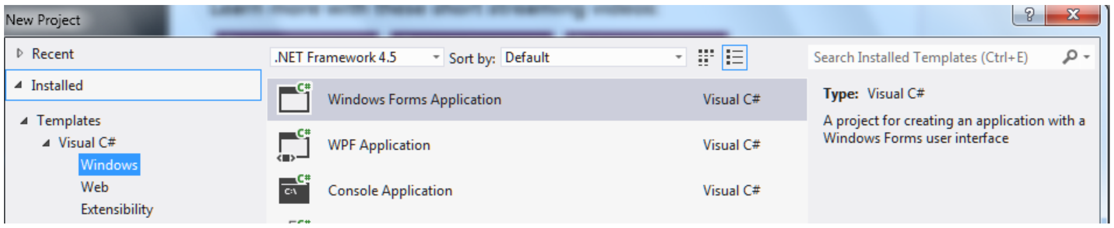
A **New Project** dialog box appears.

Step3: In the Installed Templates tree at the left of the screen, select the type of project you are creating:

Visual C#:

Visual C#: Windows

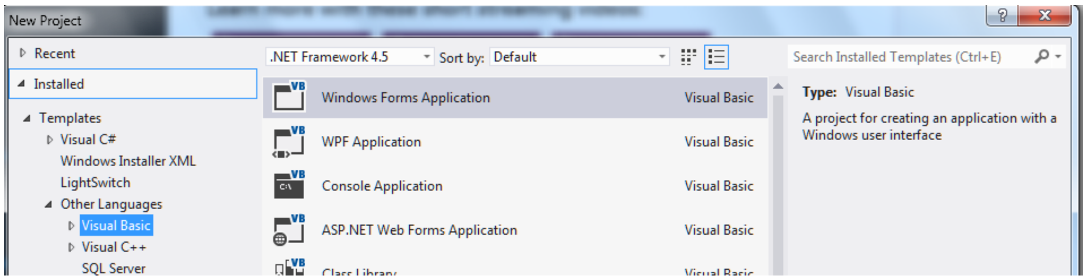
Select **Windows Forms Application**.


New Project
Recent
Installed
Templates
Visual C#
Windows
Web
Extensibility
.NET Framework 4.5
Sort by: Default
Search Installed Templates (Ctrl+E)
Windows Forms Application Visual C#
WPF Application Visual C#
Console Application Visual C#
Type: Visual C#
A project for creating an application with a Windows Forms user interface

Visual Basic:

Other Languages: Visual Basic: Windows

Select **Windows Forms Application**.


New Project
Recent
Installed
Templates
Visual C#
Windows Installer XML
LightSwitch
Other Languages
Visual Basic
Visual C++
SQL Server
.NET Framework 4.5
Sort by: Default
Search Installed Templates (Ctrl+E)
Windows Forms Application Visual Basic
WPF Application Visual Basic
Console Application Visual Basic
ASP.NET Web Forms Application Visual Basic
Class Library Visual Basic

Type: Visual Basic

A project for creating an application with a Windows user interface

Step4: In the Name field, enter the appropriate name.

Visual C#:

```
PERS_Connect_CS
```

Visual Basic:

```
PERS_Connect_VB
```

The abbreviation **CS** indicates C# projects and VB indicates Visual Basic projects.

Step5: In Location, enter the directory where you want to save the files.

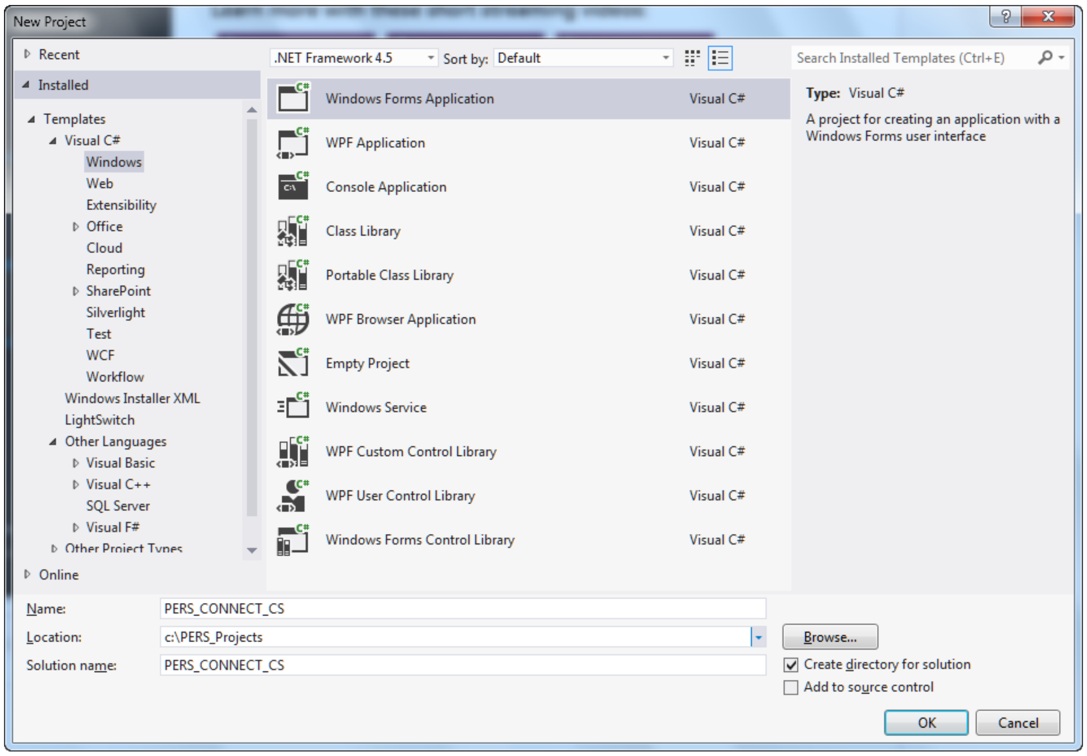
For this guide, enter this directory

```
C:\PERS_Projects
```

Step6: In Solution Name, the appropriate name, **PERS_Connect_CS** or **PERS_Connect_VB** should appear.

A solution can contain several projects; when it contains only one project, you can use the same name for both.

Step7: Check **Create directory for solution.**
Step8: Click **OK.**


New Project
Recent
Installed
Templates
Visual C#
Windows
Web
Extensibility
Office
Cloud
Reporting
SharePoint
Silverlight
Test
WCF
Workflow
Windows Installer XML
LightSwitch
Other Languages
Visual Basic
Visual C++
SQL Server
Visual F#
Other Project Types
Online
Windows Forms Application Visual C#
WPF Application Visual C#
Console Application Visual C#
Class Library Visual C#
Portable Class Library Visual C#
WPF Browser Application Visual C#
Empty Project Visual C#
Windows Service Visual C#
WPF Custom Control Library Visual C#
WPF User Control Library Visual C#
Windows Forms Control Library Visual C#
Search Installed Templates (Ctrl+E)
Browse...
OK
Cancel

.NET Framework 4.5 Sort by: Default

Type: Visual C#

A project for creating an application with a Windows Forms user interface

Name: PERS_CONNECT_CS

Location: c:\PERS_Projects

Solution name: PERS_CONNECT_CS

☑ Create directory for solution

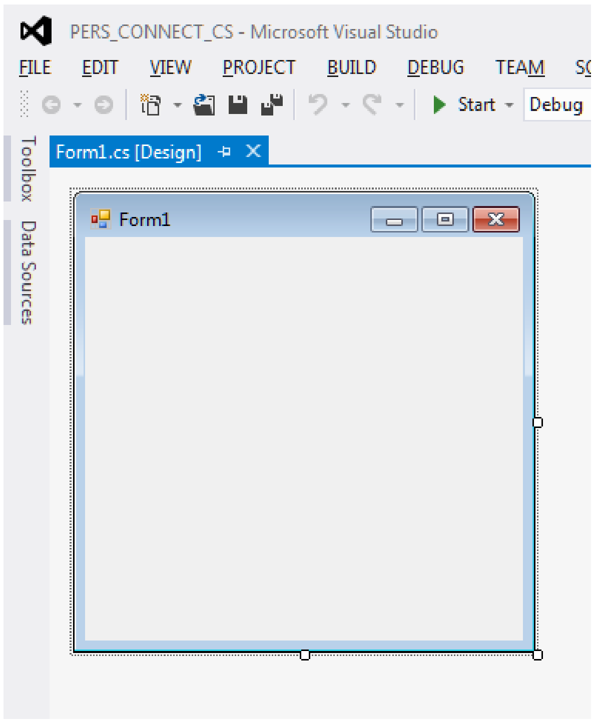
☐ Add to source control

The project is created.

The main window now displays a new title, either **PERS_Connect_CS - Microsoft Visual Studio** or **PERS_Connect_VB - Microsoft Visual Studio**, depending on the language, and contains Form1 shown below.

It is important to remember that many projects automatically name the first form Form1. This is the name of the form control. Do not confuse this with the actual name given to the code file, which is typically `Form1.cs` or `Form1.vb`.

Both Form1 and `Form1.xx` can be renamed. For the purposes of this guide, we will rename `Form1.xx` several times.

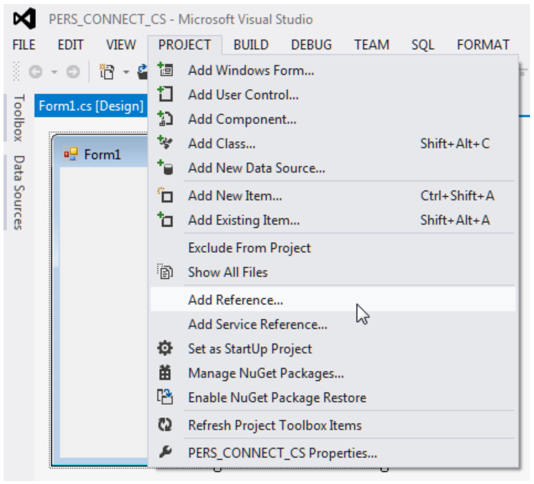

PERS_CONNECT_CS - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM
Start Debug
Toolbox
Data Sources
Form1.cs [Design]
Form1

4.2 Adding a Reference

This section shows you how to add a reference to the `Oracle.DataAccess.Rdb.dll` file, which contains the data provider, Oracle Rdb Data Provider for .NET.

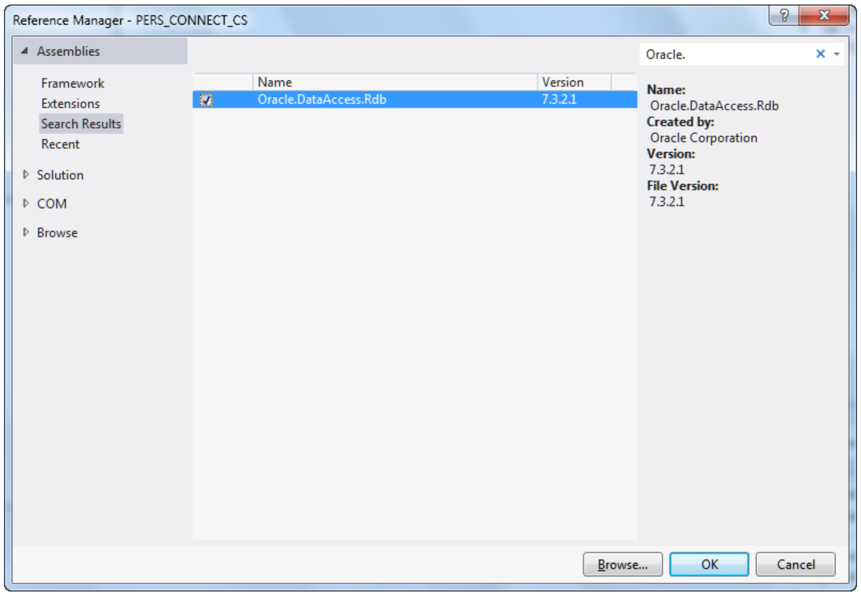
To add a reference:

Step1: From the **Project** menu, select **Add Reference**.



The Add Reference windows appears.

Step2: In the **Reference Manager** window, in the **Search Assemblies** field, Enter "Oracle." And select **Oracle.DataAccess.Rdb** that is displayed by clicking the box to the left of the name. Click OK.

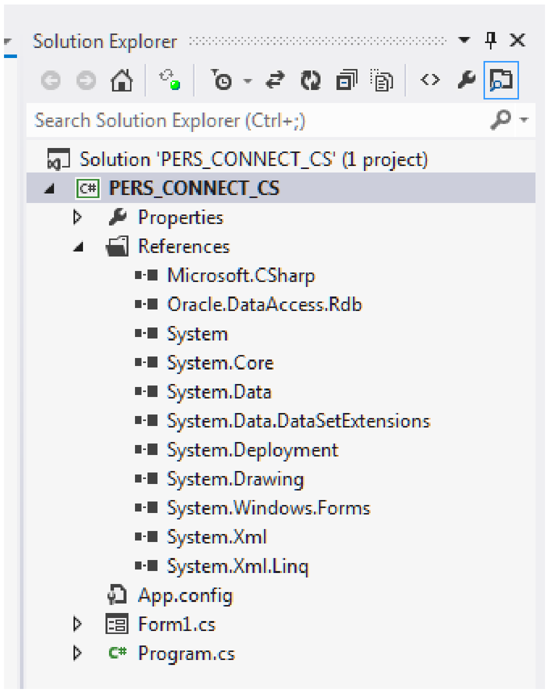


Oracle.

Name	Version
Oracle.DataAccess.Rdb	7.3.2.1

Name:
Oracle.DataAccess.Rdb
Created by:
Oracle Corporation
Version:
7.3.2.1
File Version:
7.3.2.1

Note that the new reference appears in the Solution Explorer.



Solution 'PERS_CONNECT_CS' (1 project)
PERS_CONNECT_CS
Properties
References
Microsoft.CSharp
Oracle.DataAccess.Rdb
System
System.Core
System.Data
System.Data.DataSetExtensions
System.Deployment
System.Drawing
System.Windows.Forms
System.Xml
System.Xml.Linq
App.config
Form1.cs
Program.cs

4.3 Adding Namespace Directives

You can add Oracle namespace directives that allow you to indicate an assembly's namespaces within the module. To do this, add C# `using` statements or Visual Basic `Imports` statements, at or near the top of a code file.

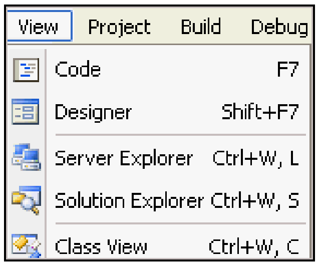
Note:

Adding a reference makes the namespace available within the application. Adding a namespace directive within the application code makes the namespace more visible and allows for additional scoping.

To add Oracle Rdb namespace directives:

Step1: With Form1 active, from the **View** menu select **Code**.

Alternatively, you can use the **F7** keyboard shortcut.

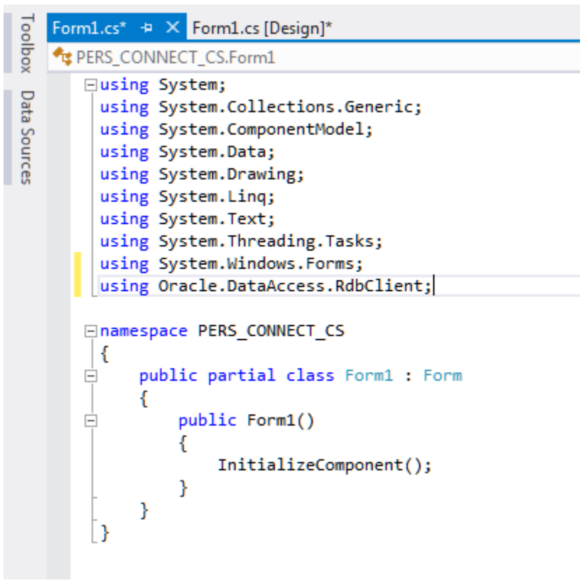


Step2: Add the following statements to the list of declarations depending on the language you are using.

Visual C#:

Add with other `using` statements, before the namespace.

```
using Oracle.DataAccess.RdbClient;
```

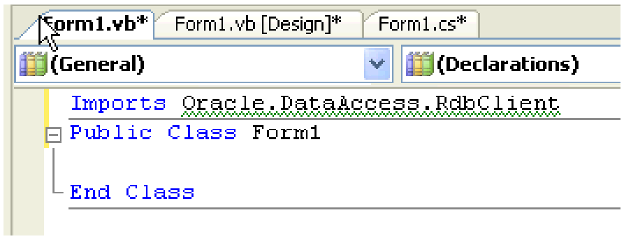


```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Oracle.DataAccess.RdbClient;

namespace PERS_CONNECT_CS
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
    }
}
```

Visual Basic:

Add to the top of the file, in the declarations section.

```
Imports Oracle.DataAccess.RdbClient
```



```
Imports Oracle.DataAccess.RdbClient
Public Class Form1

End Class
```

Step3: Save the changes by selecting **Save** from the **File** menu, or using the **Ctrl+S** keyboard shortcut.

4.4 Designing the User Interface

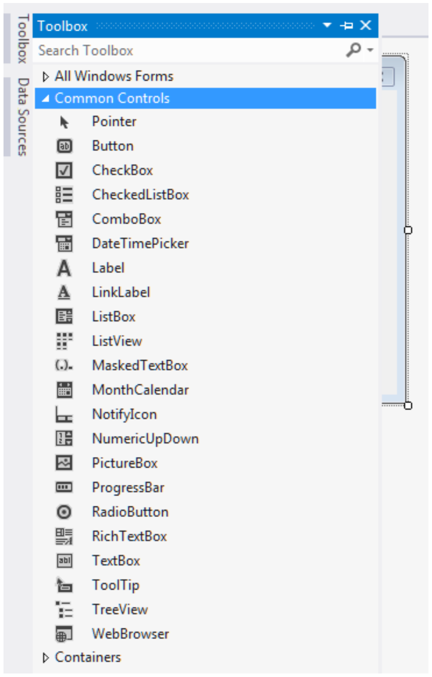
You can create a user interface by adding the toolbox controls to the design form. This interface accepts connection information from the user.
To add toolbox controls:

Step1: From the **View** menu, select **Designer**.

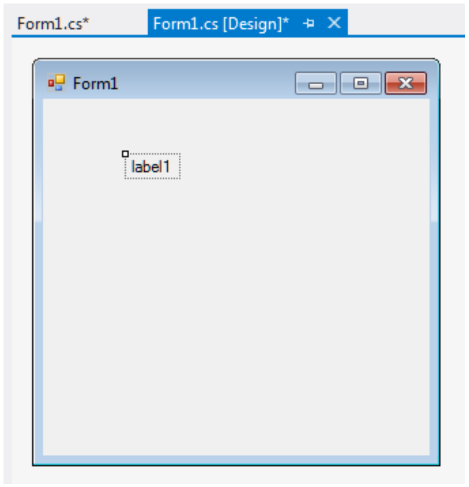
This opens Form1, in design view, if it is not already open.

You will toggle between Code and Designer a lot. The keyboard shortcuts are **F7** and **shift- F7** respectively.

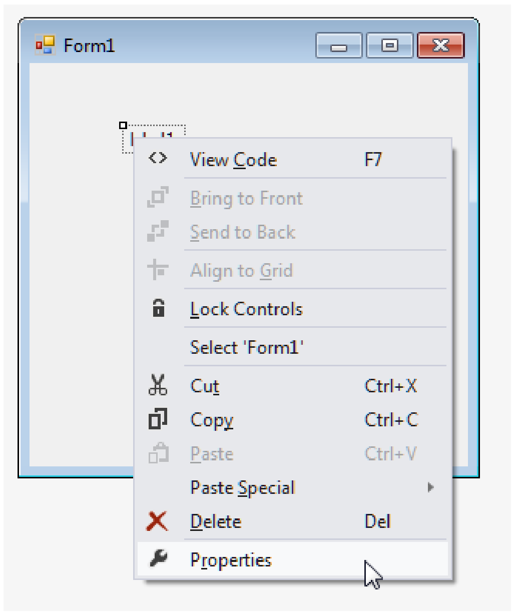
Step2: From the **View** menu, select **Toolbox**.
Step3: In the Toolbox, expand **Common Controls**.



Step4: In the Toolbox, select **Label**, and drag it onto the Form1.



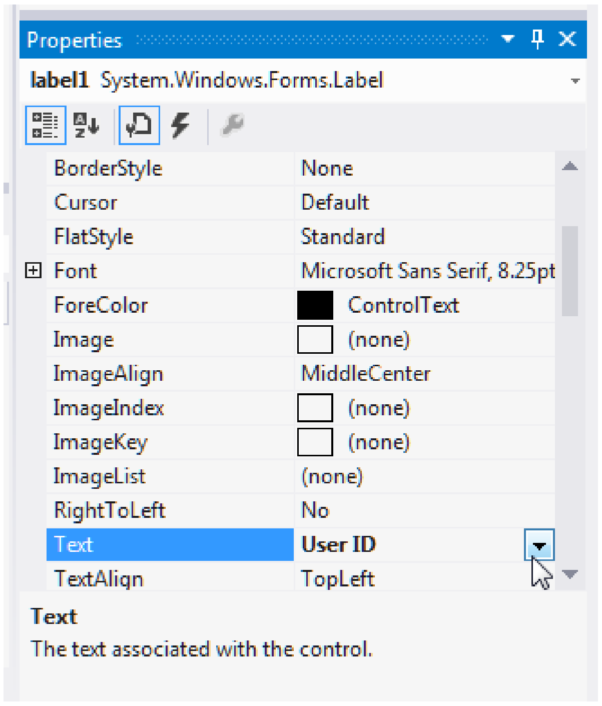
Step5: On Form1, right-click **label1**.



Step6: From the menu, select **Properties**, if the Properties Window is not already visible.

The Properties Window appears.

Step7: In the Properties Window, change the Text property from **label1** to **User ID**.

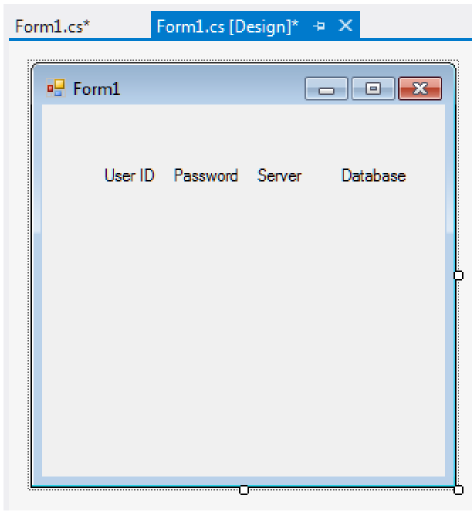


label1 System.Windows.Forms.Label

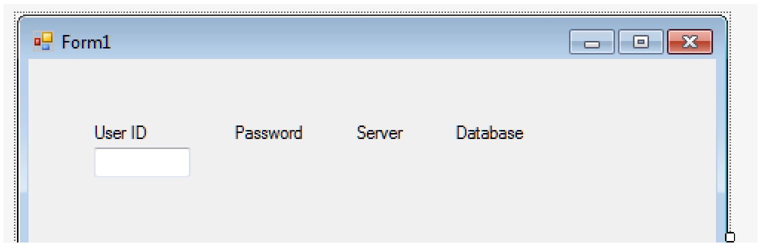
BorderStyle	None
Cursor	Default
FlatStyle	Standard
⊞ Font	Microsoft Sans Serif, 8.25pt
ForeColor	■ ControlText
Image	□ (none)
ImageAlign	MiddleCenter
ImageIndex	□ (none)
ImageKey	□ (none)
ImageList	(none)
RightToLeft	No
Text	**User ID**
TextAlign	TopLeft

Text
The text associated with the control.

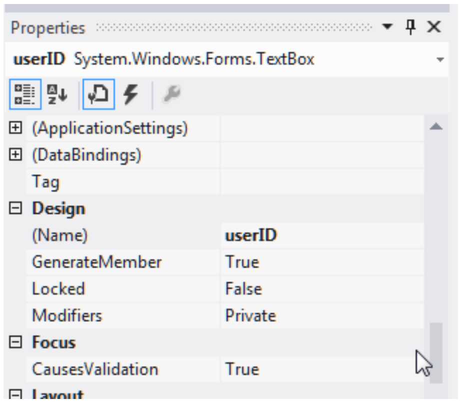
Step8: Repeat steps 4 through 7 three times, placing three more labels on Form 1 and changing their text properties to **Password**, **Server** and **Database**.



Step9: You may want to now expand the form by dragging the left margin to allow for reasonably sized fields to be accommodated. In the Toolbox, select **TextBox**, and drag it onto the Form1, under the User ID label.



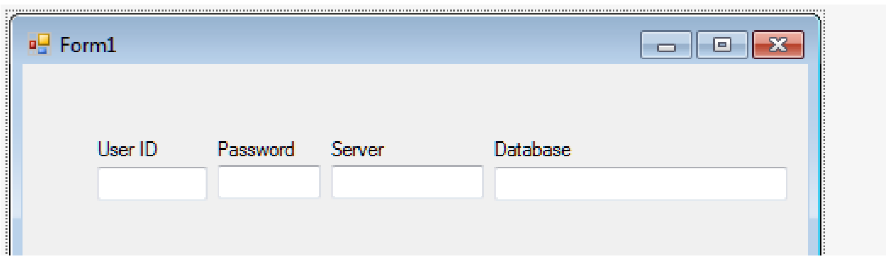
Step10: In the Properties Window, change the Name property to userID.



userID System.Windows.Forms.TextBox

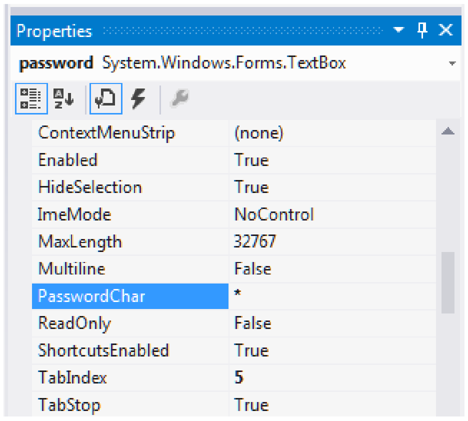
⊞ (ApplicationSettings)	
⊞ (DataBindings)	
Tag	
⊟ **Design**	
(Name)	**userID**
GenerateMember	True
Locked	False
Modifiers	Private
⊟ **Focus**	
CausesValidation	True
⊟ **Layout**	

Step11: Repeat steps 9 and 10 three times, positioning three more text boxes under the existing labels, and setting the Name property to **password**, **server** and **database**.



Step12: Select the text box under the Password label. In the Properties Window, scroll to the PasswordChar property and set it to an asterisk (*****).

This masks the password during entry.

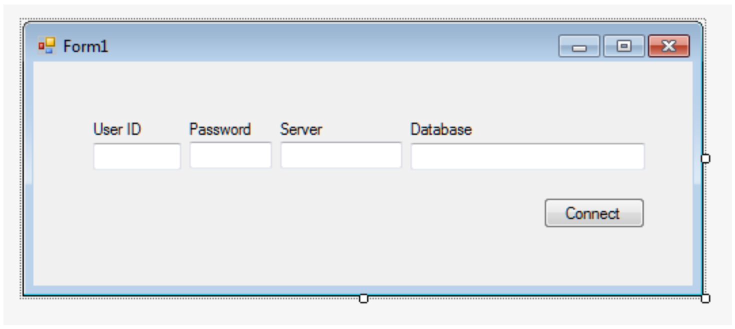


password System.Windows.Forms.TextBox

ContextMenuStrip	(none)
Enabled	True
HideSelection	True
ImeMode	NoControl
MaxLength	32767
Multiline	False
PasswordChar	*
ReadOnly	False
ShortcutsEnabled	True
TabIndex	5
TabStop	True

Step13: From the Toolbox, select **Button** and drag it onto Form1.

In the Properties Window, change the Text property of the button from **button1** to **Connect**, and change the **Name** property to **connect**.



Step14: Save.

4.5 Writing the Connection Code

Now we write the code that takes the information provided to the user interface and connects to the database.

To connect to the database, you must create a connection object.

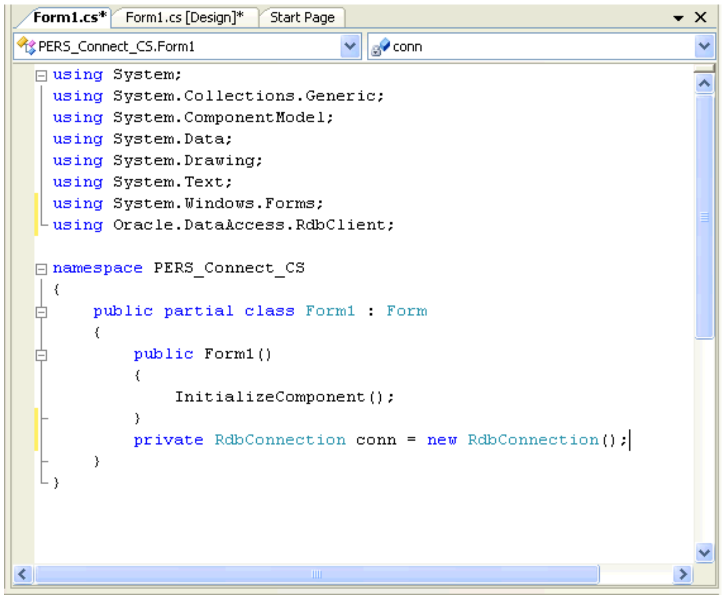
To write code that connects to the database:

These steps enable your application to connect to the database based on data that the user enters into the Form1 control. See Compiling and Running the Application.
From the **View** menu, select **Code**.

Step1: Add the code indicated to instantiate a database connection string.

Visual C#: Add the class variable `conn` to the `Form1` class right after the `public Form1()` block with this code.

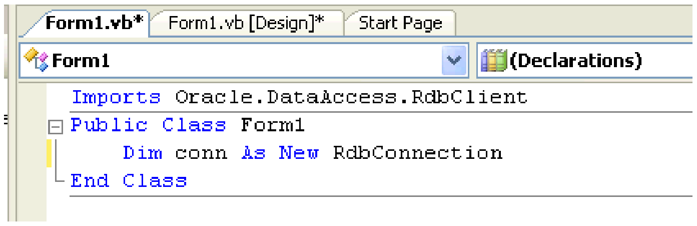
```
private RdbConnection conn = new RdbConnection();
```



```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Oracle.DataAccess.RdbClient;

namespace PERS_Connect_CS
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private RdbConnection conn = new RdbConnection();
    }
}
```

Visual Basic: Add the `conn` class variable in the `Form1` class declaration, using this code.

```
Public Class Form1
   Dim conn As New RdbConnection
```



```
Imports Oracle.DataAccess.RdbClient
Public Class Form1
    Dim conn As New RdbConnection
End Class
```

Step2: Save your changes.

Step3: Change to Designer view by clicking on the **View** menu and selecting **Designer**.

Step4: Double-click the **Connect** button on Form1 to open the code window to the `connect_Click()` method.

Insert the code indicated into the `connect_Click()` method.

Visual C#:

```
conn.ConnectionString = "User Id=" + userID.Text +
  ";pwd=" + password.Text +
  ";server=" + server.Text +
  ";database=" + database.Text +
  ";type=Thin;";

conn.Open();
```

Visual Basic:

```
conn.ConnectionString = "User Id=" + userID.Text & _
  ";pwd=" + password.Text & _
  ";server=" + server.Text & _
  ";database=" + database.Text & _
  ";type=Thin;"

conn.Open()
```

Visual Basic `Imports` statements, at or near the top of a code file.

Before a connection can be opened, it must be built from user input for the `User Id`, `pwd`, `server`, and `database`.
The `Open()` method makes the actual connection.

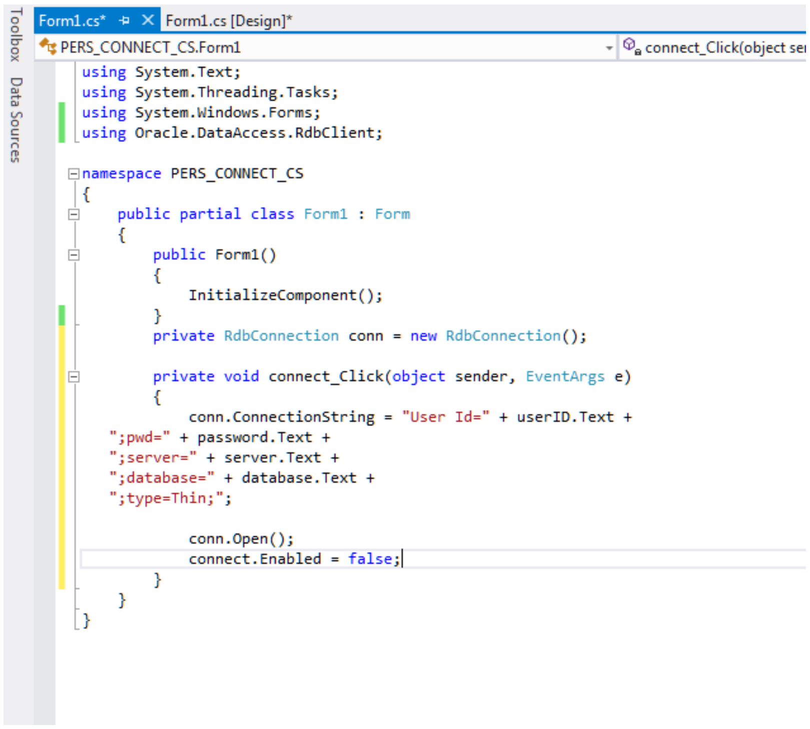
Note that in this example we are making a connection to a JDBC Thin Server as designated by the ";type=Thin;" portion of the connection string we are building. If you wish to make a connection to a SQL/Services service, omit this part of the connection string.

Step5: Set the `Enabled` attribute of the button to `false` by inserting the indicated code at the end of the `connect_Click()` method.

This disables the `Connect` button, which is a good practice once a connection is successfully made.

Visual C#:

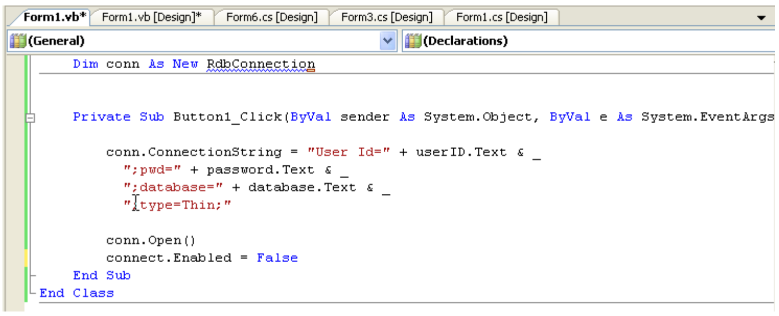
```
connect.Enabled = false;
```



```
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Oracle.DataAccess.RdbClient;

namespace PERS_CONNECT_CS
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private RdbConnection conn = new RdbConnection();

        private void connect_Click(object sender, EventArgs e)
        {
            conn.ConnectionString = "User Id=" + userID.Text +
   ";pwd=" + password.Text +
   ";server=" + server.Text +
   ";database=" + database.Text +
   ";type=Thin;";

            conn.Open();
            connect.Enabled = false;
        }
    }
}
```

Visual Basic:

```
connect.Enabled = false
```



```
    Dim conn As New RdbConnection


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs

        conn.ConnectionString = "User Id=" + userID.Text & _
          ";pwd=" + password.Text & _
          ";database=" + database.Text & _
          ";type=Thin;"

        conn.Open()
        connect.Enabled = False
    End Sub
End Class
```

You have now finished writing an application that can connect to the Oracle Rdb database.

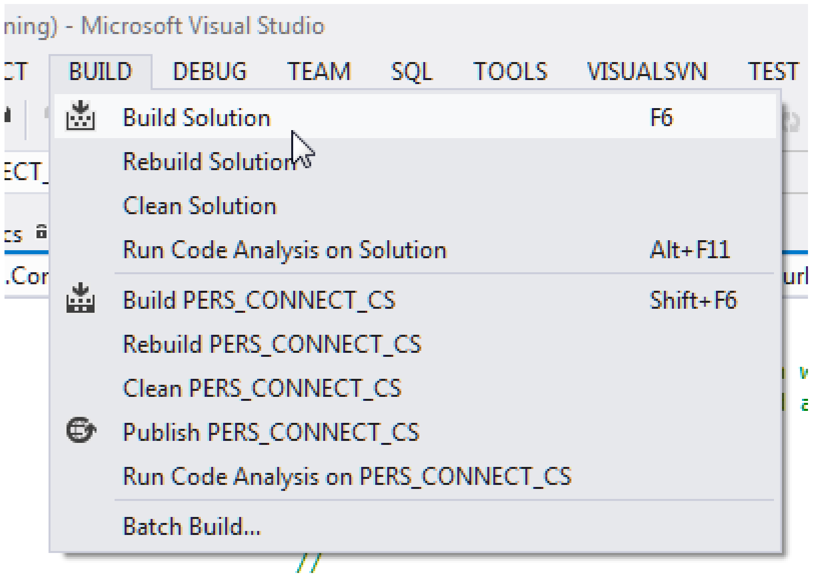
The following sections show how to use it.

4.6 Compiling and Running the Application

This section shows how to compile and run the application you created in the previous sections.

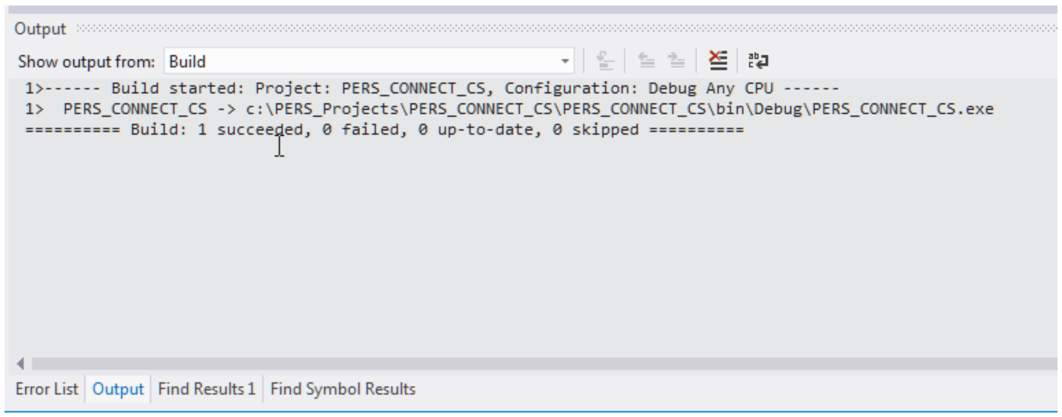
To compile and run the application:

Step1: From the **Build** menu, select **Build Solution**.



Step2: Ensure that there are no errors reported in the output window, available from the **View** menu.

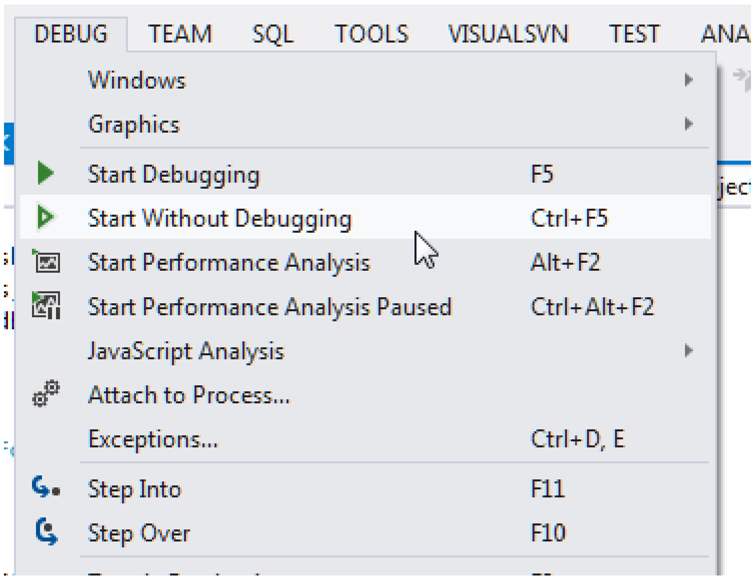
The following graphics shows a typical output result.



```
1>------ Build started: Project: PERS_CONNECT_CS, Configuration: Debug Any CPU ------
1>  PERS_CONNECT_CS -> c:\PERS_Projects\PERS_CONNECT_CS\PERS_CONNECT_CS\bin\Debug\PERS_CONNECT_CS.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

Step3: If there are any errors indicated, from the **View** menu, select **Error List** and fix the errors.

Step4: From the **Debug** menu, select **Start Without Debugging** to run the application.



Step5: In the Form1 application, enter the User ID, Password , Server, and Database. If you used the connection string exactly as shown in **Step 4** above, you will be making a connection to a JDBC Thin Server. Please ensure that the data you enter in the **Server** field does contain a valid JDBC server specified in the form "`host:port`", for example "`my_server_node:1701`"

If you omit the "`;type=Thin;`" part of the connection string, then you are requesting for a SQL/Services service connection, in which case ensure the data entered in the **Server** field is in the form of "`host:service`" for example, "`my_sqs_server:GENERIC`".

Step6:

Click **Connect**.

Once the connection is opened, the Connect button is disabled. You have succeeded in implementing a connection to an Oracle Rdb Database.

4.7 Error Handling

Applications must be able to handle run-time errors gracefully. For example, if you try to log in using an incorrect password, the application you developed so far cannot establish a connection to the database, and exits with the following unhandled exception

```
%RDB-E-AUTH_FAIL: authentication failed for user
```

You must reselect Start Without Debugging to try this with a different password.

Error handling manages occurrences of conditions that change the normal flow of program execution. Oracle Rdb Data Provider for .NET contains a single classes for error handling and support:

- The `RdbException` class represents an exception that is thrown when the Oracle Rdb Data Provider for .NET encounters an error. Each `RdbException` object contains a description and an error number that describes the error or warning.

4.7.1 Using Try-Catch-Finally Block Structure

.NET languages use Try-Catch-Finally block structure for error handling. With this structure, the Try code is the main code, the goal that the application wants to accomplish. The Catch code catches errors of various types, as shown in the next two section. The Finally block comes last and always executes.

The Finally block frequently contains the `Dispose` method, which closes and disposes of the connection. Having the `Dispose` method in the Finally block ensures that the database connection is always closed after the Try-Catch-Finally block completes. Closing database connections after the application no longer requires database access is important for many reasons, especially data security.

Attempting to close a closed database connection does not cause an error. The attempt is irrelevant. Nonetheless, placing `Dispose()` in the Finally code block guarantees that the connection is closed.

The next section shows how to use Try-Catch-Finally block structure with general errors, and the section after that, with Oracle Rdb errors.

4.7.2 Handling General Errors

This section shows how to handle general errors using a Try-Catch-Finally block.

To handle general errors:

Step1: Change the code of the `connect_Click()` method in `Form1` by adding an implementation of the Try-Catch-Finally syntax.

New code is in bold font.

Visual C#:

```
private void connect_Click(object sender, EventArgs e)
{
     conn.ConnectionString = "User Id=" + userID.Text +
   ";pwd=" + password.Text +
   ";server=" + server.Text +
   ";database=" + database.Text +
   ";type=Thin;";conn.Open();

  try
  {
    conn.Open();
    connect.Enabled = false;
  }
  catch (Exception ex)
  {
    MessageBox.Show(ex.Message.ToString());
  }
  finally
  {
    conn.Dispose();
  }
}
```

Alternatively, you can use C# syntax that disposes of a connection when it goes out of scope, with the `using` keyword, as follows:

```
using (RdbConnection conn = new RdbConnection())
{
  conn.Open();
  // application code
  ...
}
```

Visual Basic:

```
Try
  conn.Open()
  connect.Enabled = false

Catch ex As Exception
  MessageBox.Show(ex.Message.ToString())
```

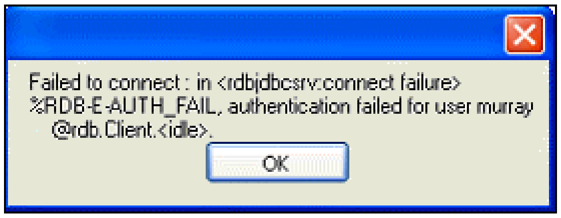
```
Finally
  conn.Dispose()
End Try
```

Step2: From the Build menu, select Rebuild Solution.

Ensure that there are no errors.

Step3: From the Debug menu, select Start Without Debugging.
Step4: Run the application again, as described in section Compiling and Running the Application, and attempt to connect using an incorrect password.

This time, the application catches the error and displays it in a pop-up window, %RDB-E-AUTH_FAIL: `authentication failed for user`.



```
Failed to connect : in <rdbjdbcsrv:connect failure>
%RDB-E-AUTH_FAIL, authentication failed for user murray
   @rdb.Client.<idle>.
```

Show Navigation

Chapter 5 Retrieving and Updating with ORDP.NET

This chapter contains:

5.1 Using the Command Object

To view, edit, insert or delete data in a database, you must encapsulate a request in an `RdbCommand` object specifying a SQL command, stored procedure, or table name. The `RdbCommand` object creates the request, sends it to the database, and returns the result.

To use the command object:

Step1: Make two copies of `Form1.xx`, from application PERS_Connect_`xx` in Building a Simple .NET Application Using ORDP.NET. To make copies, see the instructions in Copying a Form.

Name the copies `Form2.cs` or `Form2.vb` and `Form3.cs` or `Form3.vb`. The first copy is for the first part of the chapter, and the second copy for the second part of the chapter

Step2: Open `Form2.cs` or `Form2.vb`.

Note that the actual form in the designer still says Form1, as you renamed code files but not the actual form controls within the project.

Step3: Create a string that represents the SQL query and add to the body of the `try` statement.

The new code is in bold typeface.

Visual C#:

```
try
{
    conn.Open();
    connect.Enabled = false;

    // SQL Statement
    string sql = "select department_name from departments"
       + " where department_code = 'ENG'";
}
```

Visual Basic:

```
Try
      conn.Open()
      connect.Enabled = False

     Dim sql As String = "select department_name from departments" &
_
       "where department_code = 'ENG'"
```

Step4: Use the new `sql` variable to create the **`RdbCommand`** object, and set the **`CommandType`** property to run a text command.

Visual C#:

```
try
{
    conn.Open();
    connect.Enabled = false;

    // SQL Statement
    string sql = "select department_name from departments"
       + " where department_code = 'ENG'";

   RdbCommand cmd = new RdbCommand(sql, conn);
   cmd.CommandType = CommandType.Text;
}
```

Visual Basic:

```
Try
    conn.Open()
    connect.Enabled = False

    Dim sql As String = "select department_name from departments" & _
         "where department_code = 'ENG'"

    Dim cmd As New RdbCommand(sql, conn)
    cmd.CommandType = CommandType.Text
```

Step5: Save your work.

5.2 Retrieving Data: a Simple Query

This section demonstrates retrieving data from the database.

The `ExecuteReader()` method of an `RdbCommand` object returns an `RdbDataReader` object, which can be accessed to display the result on the form. The application uses a **ListBox** to display the results.

To retrieve data:

Step1: Create an `RdbDataReader` object, by adding the code indicated to the bottom of the Try block of the **`connect_Click`**`()` method.

This enables you to read the result of the query.

The new code is in bold typeface

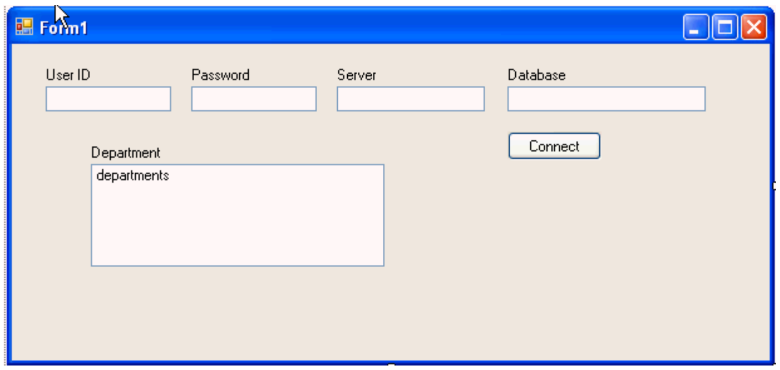
Visual C#:

```
try
{
    conn.Open();
    connect.Enabled = false;

    // SQL Statement
    string sql = "select department_name from departments"
       + " where department_code = 'ENG'";

   RdbCommand cmd = new RdbCommand(sql, conn);
   cmd.CommandType = CommandType.Text;

   RdbDataReader dr = cmd.ExecuteReader();
   dr.Read();

}
```

Visual Basic:

```
Try
    conn.Open()
    connect.Enabled = False

    Dim sql As String = "select department_name from departments" & _
         "where department_code = 'ENG'"

    Dim cmd As New RdbCommand(sql, conn)
    cmd.CommandType = CommandType.Text

    Dim dr As RdbDataReader = cmd.ExecuteReader()
    dr.Read()
```

Step2: Open Form1 in **Design view**. From the **View** menu, select **Designer**.
Step3: From the **View** menu, select **Toolbox**.
Step4: From the **Toolbox**, select a **Label** and drag it onto Form1.
Step5: From the **View** menu, select **Properties Window**.
Step6: In the **Properties** window, change the **Text** of the label to `Department`.
Step7: From the Toolbox, under Window forms, select a **ListBox** and drag it onto Form1.
Step8: In the **Properties** window, under Design, change the **Name** to `departments`.



Step9: Add accessor type methods for retrieving data from the query result.

Double-click the **connect** button to edit the **`connect_click`**`()` method, and add the code indicated to the bottom of the `try` block.

Visual C#:

```
departments.Items.Add(dr.GetString(0));
```

Visual Basic:

```
departments.Items.Add(dr.GetString(0))
```

Typed accessors, such as `GetString`, return native .NET data types and native Oracle Rdb data types. Zero-based ordinals passed to the accessors specify which column in the result set to return.

The try block should look like this now:

Visual C#:

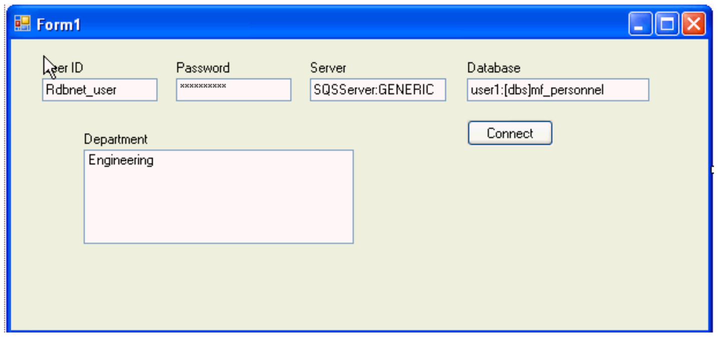
```
try
{
    conn.Open();
    connect.Enabled = false;

    // SQL Statement
    string sql = "select department_name from departments"
       + " where department_code = ‘ENG’";

   RdbCommand cmd = new RdbCommand(sql, conn);
   cmd.CommandType = CommandType.Text;

   RdbDataReader dr = cmd.ExecuteReader();
   dr.Read();

   departments.Items.Add(dr.GetString(0));

}
```

Visual Basic:

```
Try
    conn.Open()
    connect.Enabled = False

    Dim sql As String = "select department_name from departments" & _
         "where department_code = ‘ENG’"

    Dim cmd As New RdbCommand(sql, conn)
    cmd.CommandType = CommandType.Text

    Dim dr As RdbDataReader = cmd.ExecuteReader()
    dr.Read()

    departments.Items.Add(dr.GetString(0))
```

Step10: Build and save the application.
Step11: Run the application. Enter the login server and database information.

After you connect, the `departments` list box shows `Engineering`, the correct name for department with code ‘ ENG’ in the Departments table, as requested by the `SELECT` statement.



5.3 Retrieving Data: Bind Variables

Bind variables are placeholders inside a SQL statement.

The following code shows a typical `SELECT` statement that does not use bind variables, with the value `10` specified in the `WHERE` clause of the statement.

```
SELECT department_name FROM departments WHERE department_code = ‘ENG’
```

The following code replaces the numerical value with a bind variable `:department_id`. A bind variable identifier always begins with a single colon (`:`).

```
SELECT department_name FROM departments WHERE department_code =
:department_id
```

Note that bind variables can also be used with `UPDATE`, `INSERT`, and `DELETE` statements, and also with stored procedures. The following code illustrates how to use bind variables in an `UPDATE` statement:

```
UPDATE departments SET department_name = :department_name
  WHERE departname_code = : department_id
```

You can use the `RdbParameter` class to represent each bind variable in your .NET code. The `RdbParameterCollection` class contains the `RdbParameter` objects associated with the `RdbCommand` object for each statement. The `RdbCommand` class passes your SQL statement to the database and returns the results to your application.

To retrieve data using bind variables:

Step1: Move the **ListBox** named **Departments** to the right.
Step2: From the **View** menu, select **Toolbox**.
Step3: From the Toolbox, select a **TextBox** and drag it onto Form1, under the label that says **Department**.
Step4: From the **View** menu, select **Properties** Window.
Step5: In the Properties window, change **Name** to **`departmentID`**.

Step6: Change the `SELECT` statement to use the bind variable by adding the code indicated to the Try block of the `connect_Click()` method.

Changed or new code is in bold typeface.

Visual C#:

```
string sql = "select department_name from departments "+
 "where department_code = :department_id";
RdbCommand cmd = new RdbCommand(sql, conn);
cmd.CommandType = CommandType.Text;
RdbParameter p_department_id = new RdbParameter("department_id",
DbType.String);

p_department_id.Value = departmentID.Text;
cmd.Parameters.Add(p_department_id);

RdbDataReader dr = cmd.ExecuteReader();
dr.Read();

departments.Items.Add(dr.GetString(0));
```

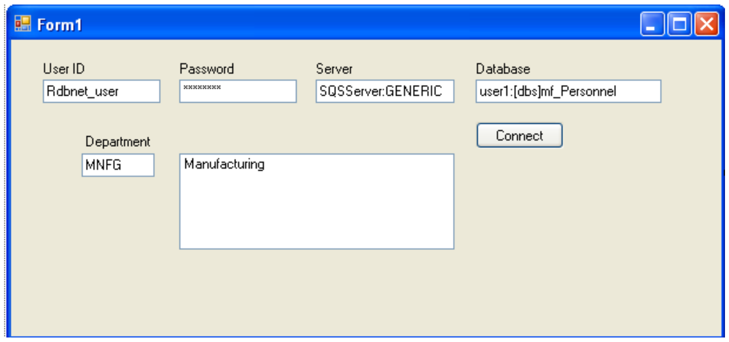
Visual Basic:

```
Dim sql As String = "select department_name from departments " & _
  "where department_code = :department_id"
Dim cmd As RdbCommand = New RdbCommand(sql, conn)
cmd.CommandType = CommandType.Text
Dim p_department_id as RdbParameter = new
RdbParameter("department_id", DbType.String)

p_department_id.Value = departmentID.Text
cmd.Parameters.Add(p_department_id)

Dim dr As RdbDataReader = cmd.ExecuteReader()
dr.Read()

departments.Items.Add(dr.GetString(0))
```

For this code, the parameter object sets the `DbType` property, but there is no need to set the `Direction` property because it uses the default value, Input. There is no need to set the Size property because the object is an input parameter, and the data provider can determine the size from the value.

Step7: **Save** and **run** the application.
Step8: Enter the login information, and a typical department code, such as "MNFG", from the MF_PERSONNEL database.
Step9: Click **Connect**.

The application returns the name of the department that corresponds to the department ID.



User ID: Rdbnet_user
Password: ********
Server: SQSServer:GENERIC
Database: user1:[dbs]mf_Personnel
Department: MNFG
Manufacturing

5.4 Retrieving Data: Multiple Values

You frequently need to retrieve more than just one value from the database. A DataReader object can retrieve values for multiple columns and multiple rows. Consider the multiple column, multiple row query in the following example:

```
SELECT department_code, department_name, manager_id  FROM departments
  WHERE department_code < 'ENG'
```

Processing multiple rows from the DataReader object requires a looping construct. Also, a control that can display multiple rows is useful. Because the RdbDataReader object is a forward-only, read-only cursor, it cannot be bound to an updatable or backward scrollable control such as Windows Forms DataGrid control. An RdbDataReader object is, however, compatible with a ListBox control.

To retrieve multiple values:

Step1: In the try block of the connect_Click() method, change the SQL query to return a multiple row result set and add a while loop to enclose the read method that displays the department names.

Visual C#:

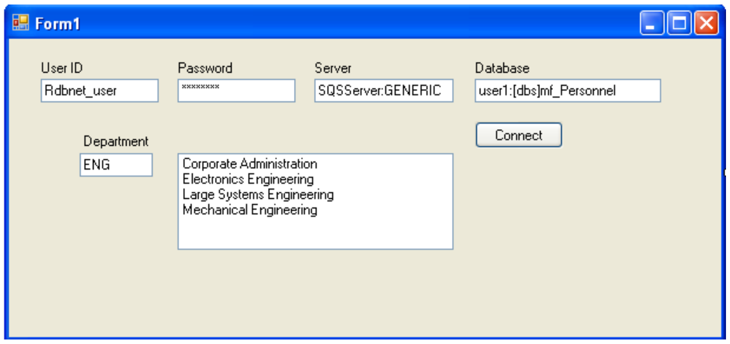
```
try
{
  ...
string sql = "select department_name from departments " +

  "where department_code < :department_id";
...
  while (dr.Read())
  {
    departments.Items.Add(dr.GetString(0));
  }
}
```

Visual Basic:

```
Try
  ...
  Dim sql As String = "select department_name from departments " & _
        "where department_code < :department_id"
...
  While (dr.Read())
    departments.Items.Add(dr.GetString(0))
  End While
```

Step2: **Save** and **run** the application.
Step3: Enter the login information and enter "**ENG**" for the department.
Step4: Click **Connect**.

The application returns the name of the departments that correspond to the query.



Rdbnet_user
****
SQSServer:GENERIC
user1:[dbs]mf_Personnel
ENG

Corporate Administration
Electronics Engineering
Large Systems Engineering
Mechanical Engineering

5.5 Using the DataSet Class with Oracle Rdb Data Provider for .NET

The DataSet class provides a memory-resident copy of database data. It consists of one or more tables that store relational or XML data. Unlike an RdbDataReader object, a DataSet is updatable and backward scrollable.

To use the DataSet class:

Step1: If you have not done so before, make another copy of the Form1 that you completed in Chapter 3, and name it Form3.vb or .cs, as described in Copying a Form

Step2: If Form1.xx does not appear in the Solution Explorer, from the Project menu, select **Show All Files**.

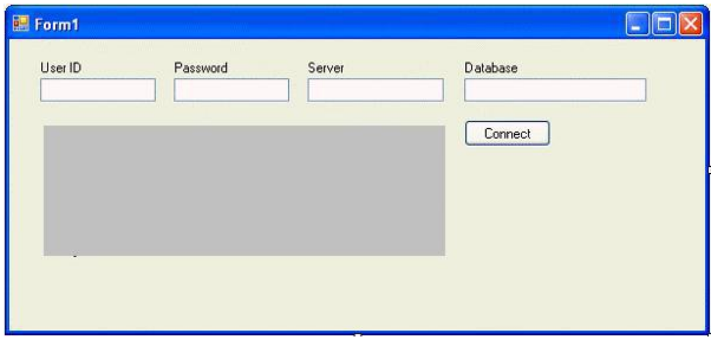
Step3: From the View menu, select **Designer view**.

Step4: From the View menu, select **Toolbox**.

Step5: From the Toolbox, select a **DataGridView** and drag it onto Form1.

Step6: From the View menu, select **Properties Window**.

Step7: In the Properties window, change the Name of the data grid view to departments.



Step8: From the View menu, select **Code.**

Step9: Immediately after the conn declaration in the code, add variable declarations to the class variables, as indicated.

Visual C#:

```
public partial class Form1 : Form
{
  public Form1()
  {
      InitializeComponent();
  }
  private RdbConnection conn = new RdbConnection();
  private RdbCommand cmd;
  private RdbDataAdapter da;
  private RdbCommandBuilder cb;
  private DataSet ds;
...
```

Visual Basic:

```
Public Class Form1
    Dim conn As New RdbConnection
    Private cmd As RdbCommand
    Private da As RdbDataAdapter
    Private cb As RdbCommandBuilder
    Private ds As DataSet
```

Step10: Within the `connect_Click()` method `try` block, add code to:

- Query the database
- Fill the `DataSet` with the result of the command query
- Bind the `DataSet` to the data grid (departments)

Visual C#:

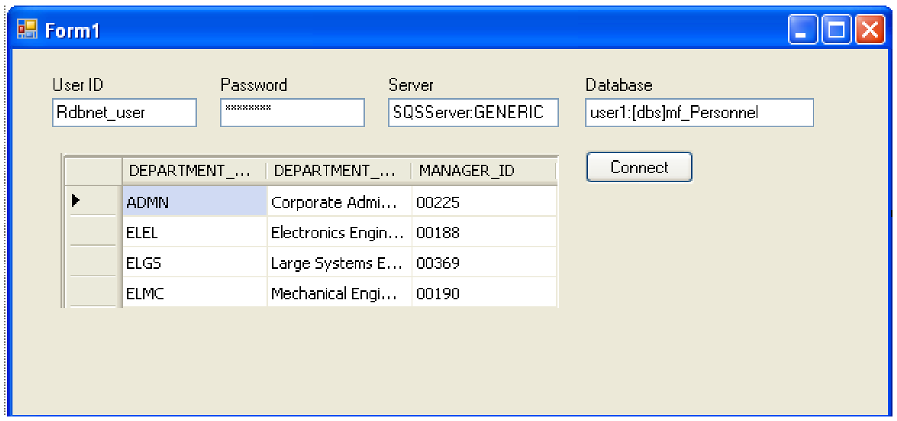
```
conn.Open();
connect.Enabled = false;

string sql = "select * from departments where "+
  "department_id < 'ENG'";
cmd = new RdbCommand(sql, conn);
cmd.CommandType = CommandType.Text;

da = new RdbDataAdapter(cmd);
cb = new RdbCommandBuilder(da);
ds = new DataSet();

da.Fill(ds);

departments.DataSource = ds.Tables[0];
```

Visual Basic:

```
conn.Open()
connect.Enabled = False

Dim sql As String = "select * from departments where " + _
 "department_id < 'ENG'"
cmd = New RdbCommand(sql, conn)
cmd.CommandType = CommandType.Text

da = New RdbDataAdapter(cmd)
cb = New RdbCommandBuilder(da)
ds = New DataSet()

da.Fill(ds)

departments.DataSource = ds.Tables(0)
```

Step11: **Build** and **save** the application.
Step12: **Run** the application, entering the **login** and **data source**.

After you successfully connect to the database, the data grid is populated with the results of the query.



User ID	Password	Server	Database
Rdbnet_user	********	SQSServer:GENERIC	user1:[dbs]mf_Personnel

DEPARTMENT_...	DEPARTMENT_...	MANAGER_ID
ADMN	Corporate Admi...	00225
ELEL	Electronics Engin...	00188
ELGS	Large Systems E...	00369
ELMC	Mechanical Engi...	00190

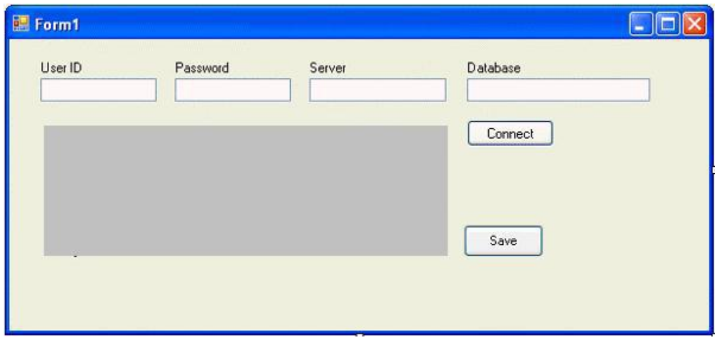
5.6 Enabling Updates to the Database

At this point, the `DataSet` contains a client copy of the database data. In this section, you will add a button that enables client data changes to be saved back to the database. The following section will show you how to test updating, inserting, and deleting the data.
To enable saving data from the DataSet to the database:

Step1: From the Toolbox, drag and drop a **Button** onto **Form1**.
Step2: In the Properties window, change the Name of the button to `save`.

Change the Text property to `Save`.

Step3: At the top of the Properties Window, click Events (the lightning bolt). In the list of events, select the click event. In the second column, enter the event name, `save_Click`.



Step4: From the View menu, select **Code**.
Step5: Add code that updates the data, to the body of the `save_Click()` method, as indicated.

Visual C#:

```
da.Update(ds.Tables[0]);
```

Visual Basic:

```
da.Update(ds.Tables(0))
```

You may see some errors show up in the Error List. These will disappear after you add the code in the next step.

Step6: Within the `Form()` method or `Form1_Load` method, add the code indicated.

Visual C#:

```
public Form1()
{
    InitializeComponent();
    save.Enabled = false;
}
```

Visual Basic:

```
Private Sub Form1_Load(ByVal sender As System.Object, & _
    ByVal e As System.EventArgs) Handles MyBase.Load
    save.Enabled = false
```

Step7: Within the `connect_Click()` method `try` block, add code to enable the Save button as indicated:

Visual C#:

```
conn.Open();
 ...
departments.DataSource = ds.Tables[0];

save.Enabled = true;
```

Visual Basic:

```
conn.Open()
...
departments.DataSource = ds.Tables(0)
```

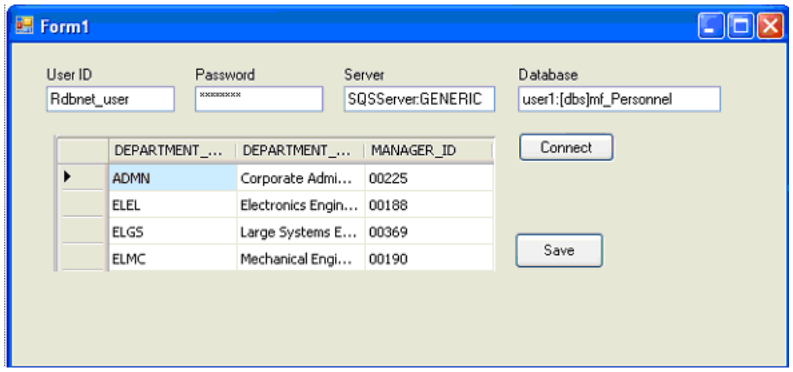
```
save.Enabled = True
```

Step8: Remove the `conn.Dispose()` call from the `finally` block in the `connect_Click()` method.

Note: In the previous code used in this example, this method was necessary to dispose or close the connection. However, with these changes to the code, it is necessary to keep the connection open after the query result returns, so that data changes made by the end user are propagated to the database. A general override call, `components.Dispose()`, is already part of the definition of Form1.

Step9: **Build** and **save** the application.
Step10: **Run** the application, entering the login and data source.

After you successfully connect to the database, the data grid is populated with the results of the query.



User ID: Rdbnet_user
Password: ********
Server: SQSServer:GENERIC
Database: user1:[dbs]mf_Personnel

DEPARTMENT_...	DEPARTMENT_...	MANAGER_ID
ADMN	Corporate Admi...	00225
ELEL	Electronics Engin...	00188
ELGS	Large Systems E...	00369
ELMC	Mechanical Engi...	00190

Show Navigation

Chapter 6 ORDP.NET and Visual Studio Wizards

In this section you will use the Visual Studio integrated development environment (IDE), along with the standard Visual Studio wizards to automatically generate the code to access data from your Rdb database

This chapter contains:

6.1 Create a New application

Create a new windows forms application using the **File →New →project** menu. Call it **empapp**.

See Creating a New Project for the steps to create a new project.

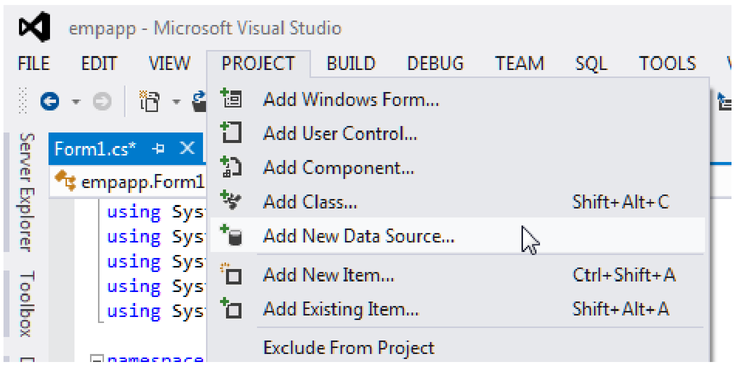
6.2 Add connection to Rdb database in Server Explorer

See Connecting to the Oracle Rdb Database for steps to connect to your database.

6.3 Create a datasource for the employees table

To create a new Data Source:

Step1: Using the top-level menu **Project → Add New** Data Source.

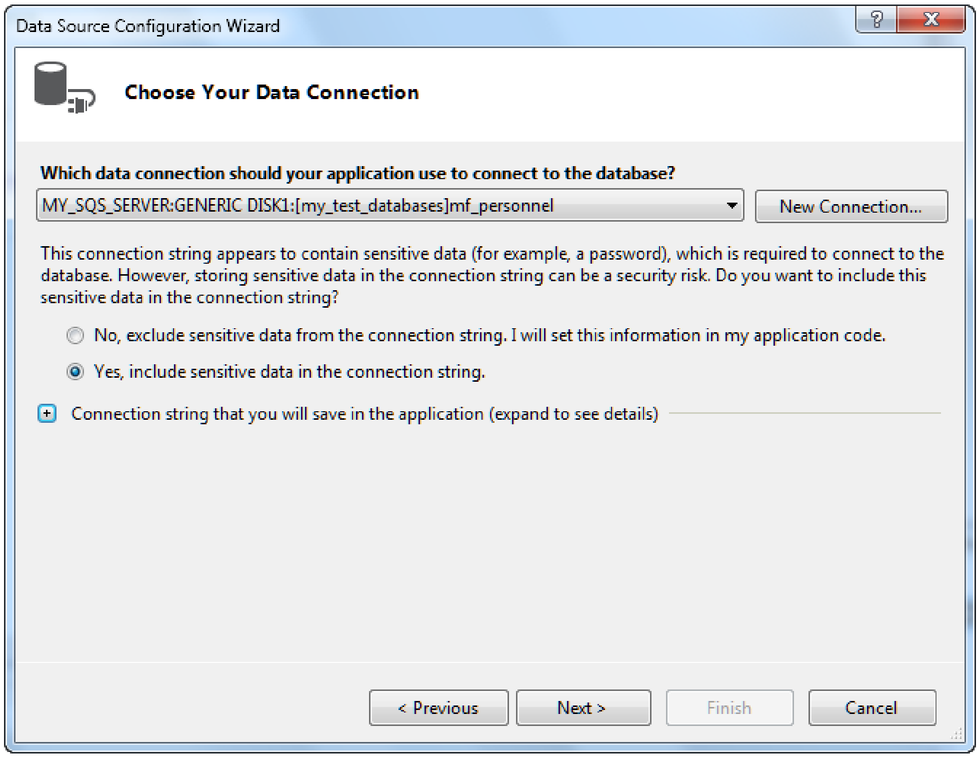


```
using Sys
using Sys
using Sys
using Sys
using Sys
namespace
```

Step2: In the wizard, keep the default selection for "Database". Click **Next**

Step3: In **Choose a Database Model** keep the default selection for "Dataset". Click **Next**

Step4: From the list of connection, select the connection to the sample that was created that was created earlier, select the option to include the sensitive data in connection string. Click **Next**



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

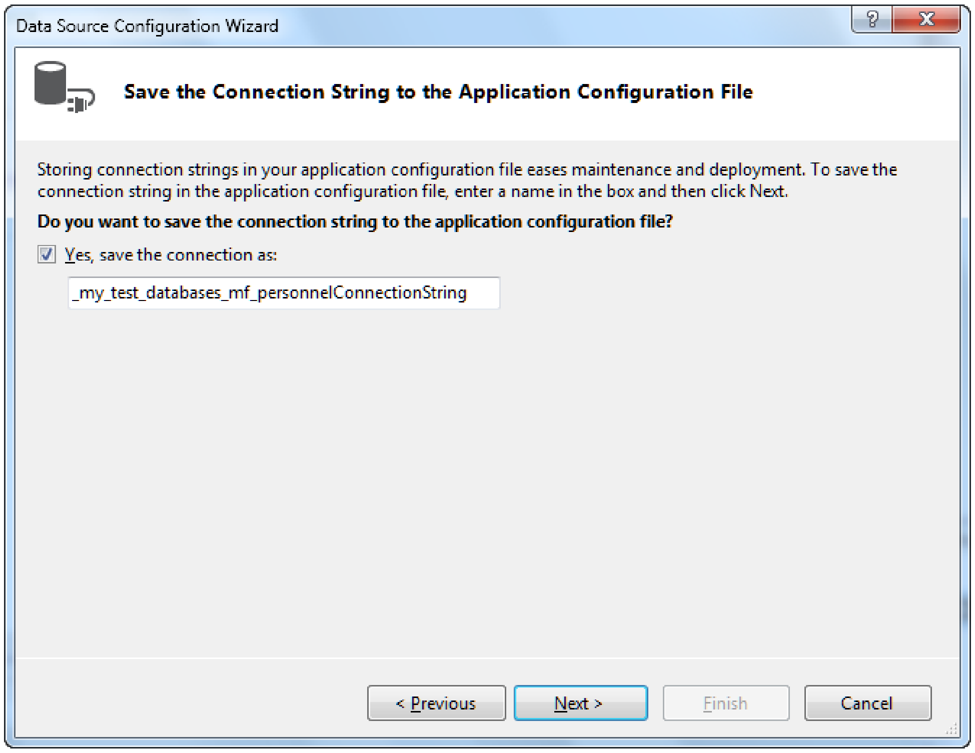
MY_SQS_SERVER:GENERIC DISK1:[my_test_databases]mf_personnel

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)

Step5: Check to save the connection string to the Application Configuration File Click **Next**


Data Source Configuration Wizard

Save the Connection String to the Application Configuration File

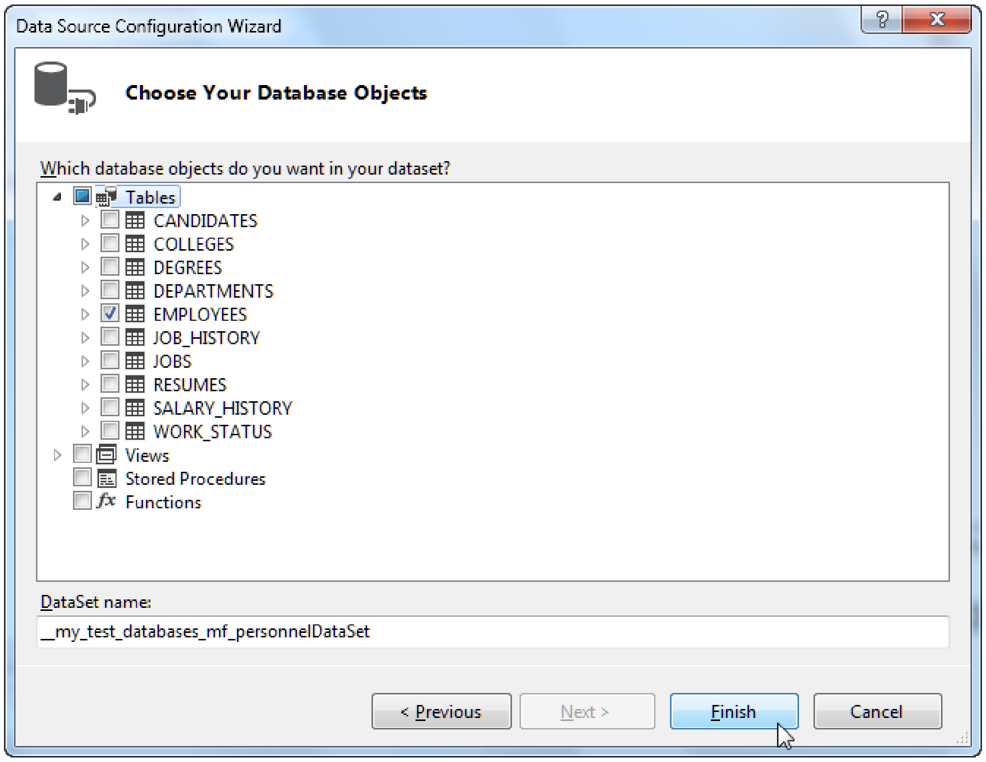
Storing connection strings in your application configuration file eases maintenance and deployment. To save the connection string in the application configuration file, enter a name in the box and then click Next.

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

[✓] Yes, save the connection as:

_my_test_databases_mf_personnelConnectionString

Step6: A tree of available table will be displayed, select the employees table from the list of tables by checking the box next to it. Click **finish**


Data Source Configuration Wizard

Choose Your Database Objects

Which database objects do you want in your dataset?

- Tables
  - CANDIDATES
  - COLLEGES
  - DEGREES
  - DEPARTMENTS
  - [✓] EMPLOYEES
  - JOB_HISTORY
  - JOBS
  - RESUMES
  - SALARY_HISTORY
  - WORK_STATUS
- Views
- Stored Procedures
- Functions

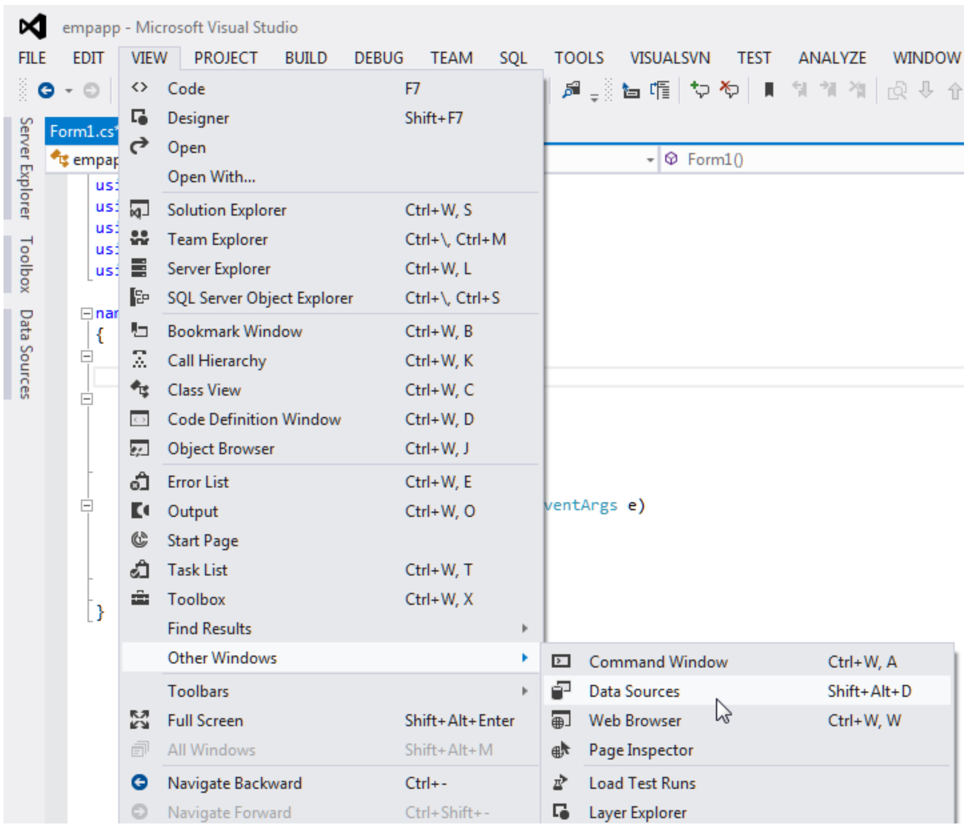
DataSet name:

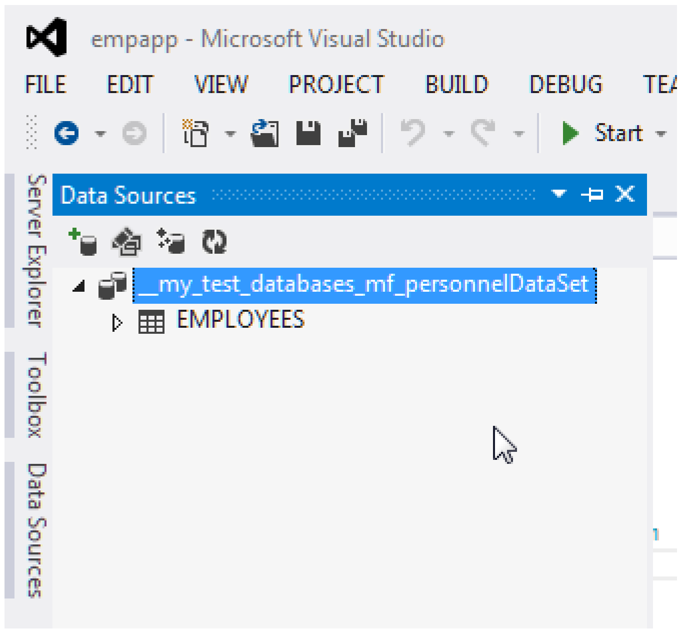
__my_test_databases_mf_personnelDataSet

Step7: A data source for employees will be added in the "Data Sources" tab panel.

You can make the data source window visible using the top level menu

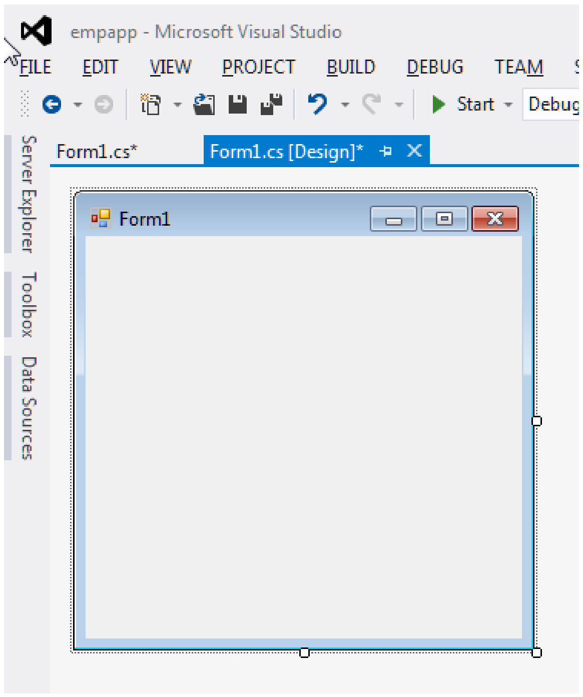
View → Other Windows→ Data sources



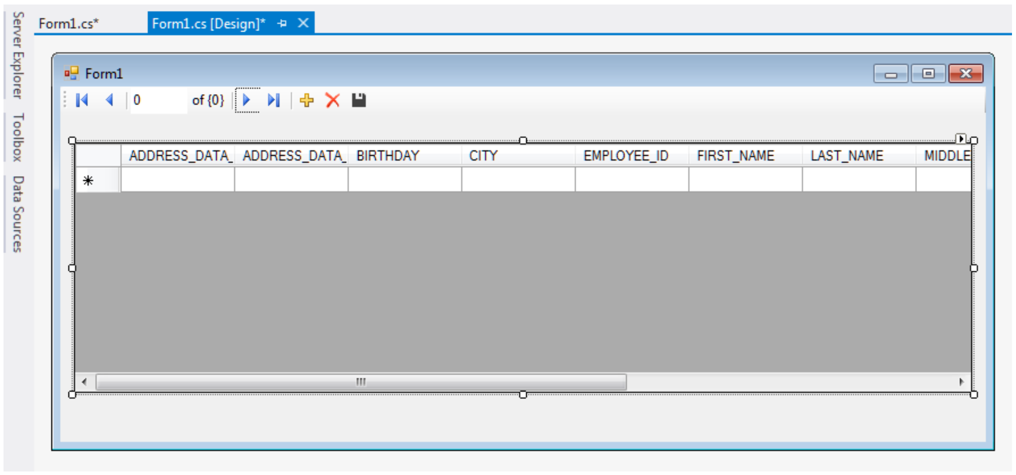

empapp - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEA
Start
Data Sources
__my_test_databases_mf_personnelDataSet
EMPLOYEES
Server Explorer
Toolbox
Data Sources

6.4 Design the windows application

Step1: Ensure that **From1.c**s is open in **design** mode.

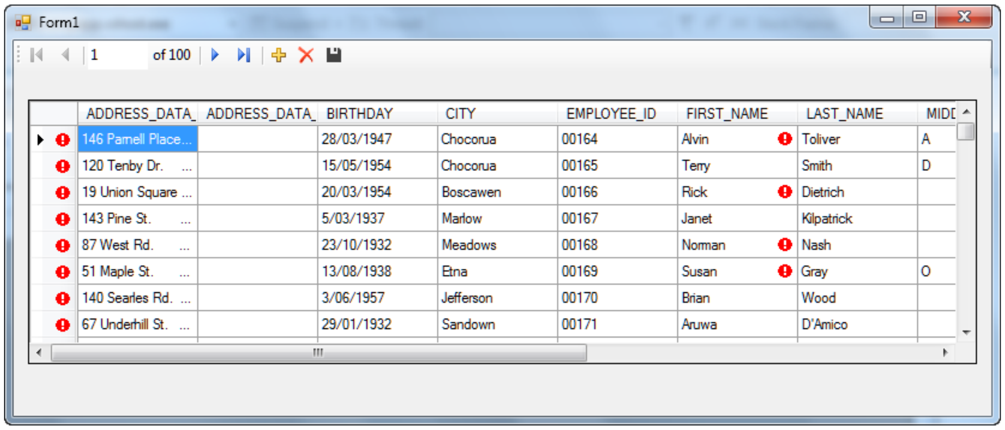


Step2: Drag and Drop the **Employees** data source from the data sources window onto the form. You will have to drag the left side of the form to make it wider so it may accommodate the fields from employees.



	ADDRESS_DATA_	ADDRESS_DATA_	BIRTHDAY	CITY	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	MIDDLE
*								

Step3: Build and Run the windows application



	ADDRESS_DATA_	ADDRESS_DATA_	BIRTHDAY	CITY	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	MIDD
▶	146 Parnell Place...		28/03/1947	Chocorua	00164	Alvin	Toliver	A
	120 Tenby Dr. ...		15/05/1954	Chocorua	00165	Terry	Smith	D
	19 Union Square ...		20/03/1954	Boscawen	00166	Rick	Dietrich	
	143 Pine St. ...		5/03/1937	Marlow	00167	Janet	Kilpatrick	
	87 West Rd. ...		23/10/1932	Meadows	00168	Norman	Nash	
	51 Maple St. ...		13/08/1938	Etna	00169	Susan	Gray	O
	140 Searles Rd. ...		3/06/1957	Jefferson	00170	Brian	Wood	
	67 Underhill St. ...		29/01/1932	Sandown	00171	Aruwa	D'Amico	

Show Navigation

Chapter 7 ORDP.NET and drag and drop

In this section you will use the Visual Studio integrated development environment (IDE), along with drag and drop to automatically generate the code to access data from your Rdb database.

This chapter contains:

7.1 Create a New application

Create a new windows application using the **File →New →project** menu. Call it **empapp2**.

See Creating a New Project for the steps to create a new project.

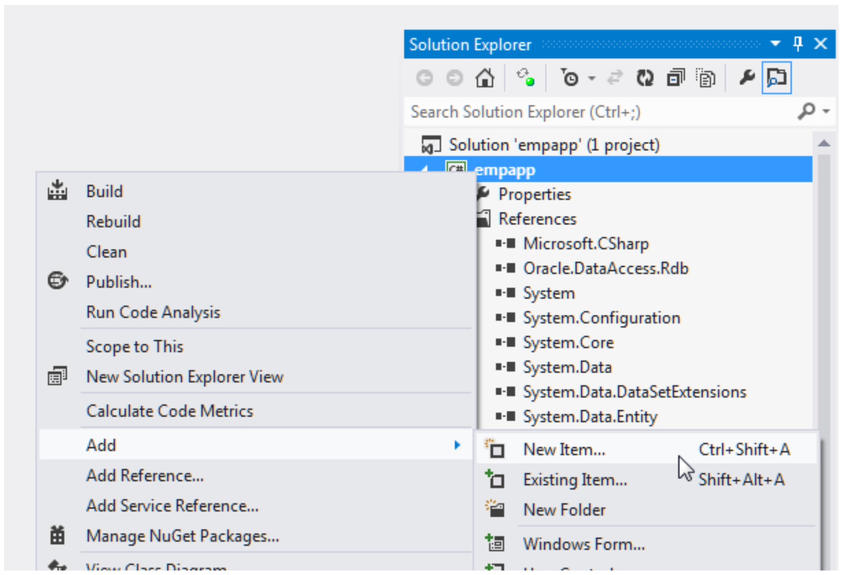
7.2 Add connection to Rdb database in Server Explorer

See Connecting to the Oracle Rdb Database for steps to connect to your database if you do not already have a connection.

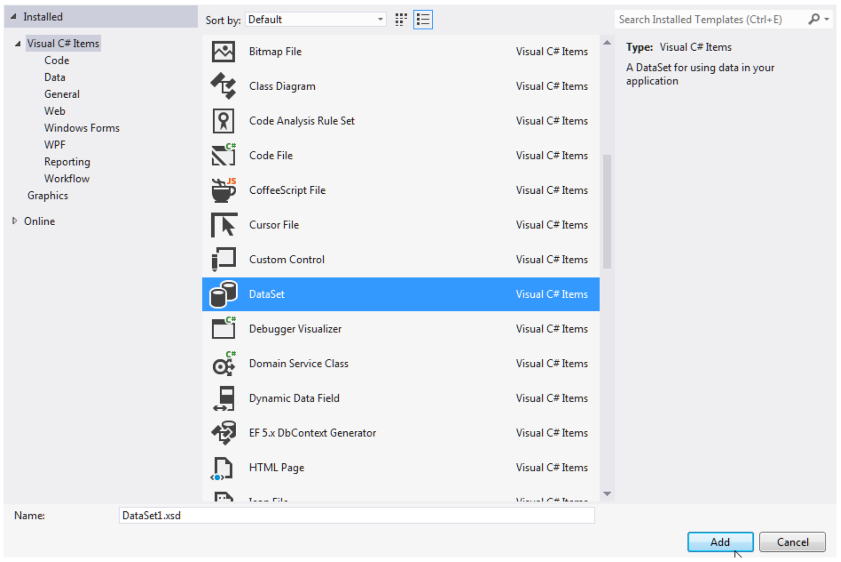
7.3 Add a new Typed DataSet

To add a new Typed DataSet to your project:

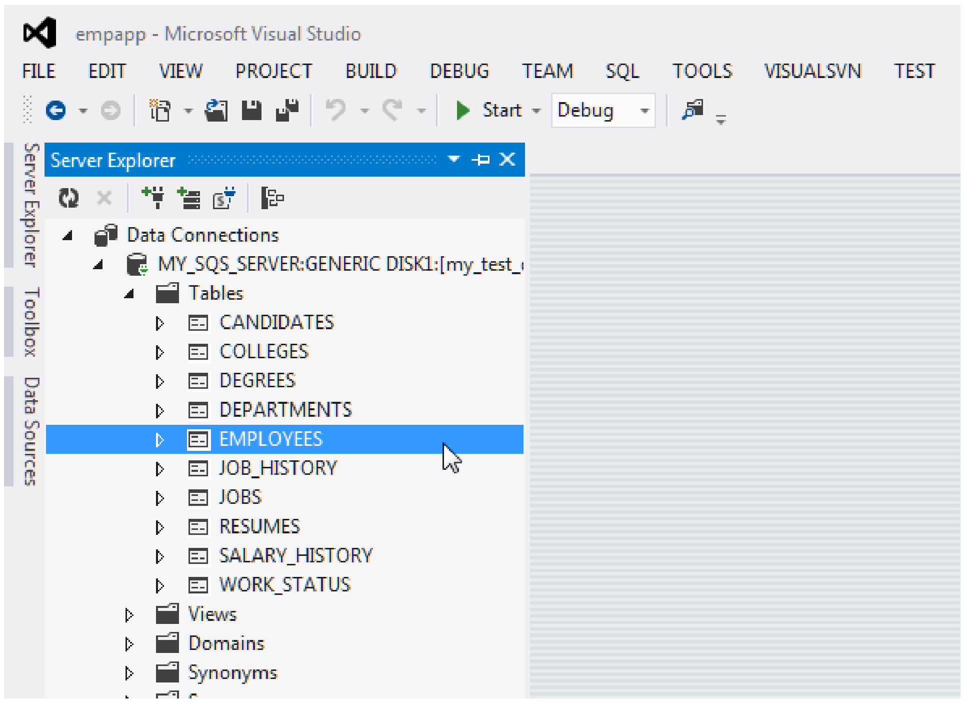
Step1: Using the Microsoft Solution Explorer, right click on the **empapp2,** and select **Add → New Item**



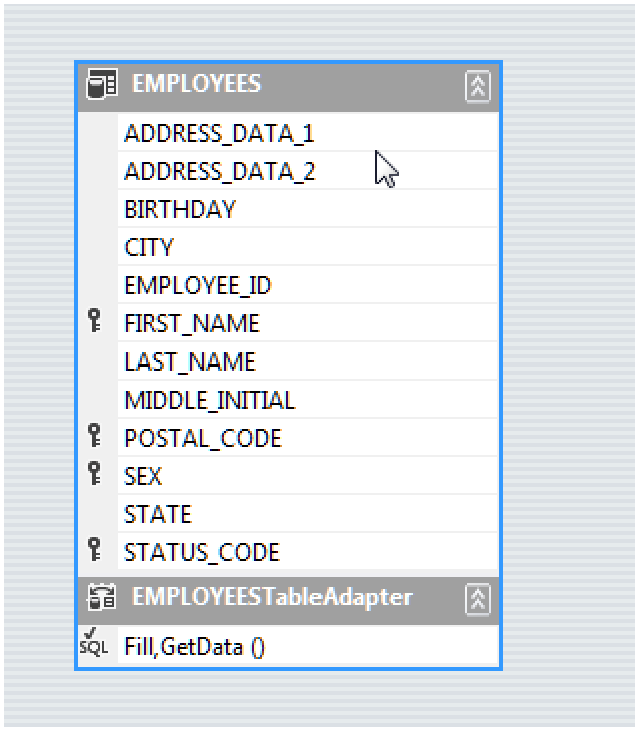
Step2: Select **DataSe**t. Click **Add**. The Microsoft Dataset Designer will be opened.


Installed
Visual C# Items
Code
Data
General
Web
Windows Forms
WPF
Reporting
Workflow
Graphics
Online
Sort by: Default
Search Installed Templates (Ctrl+E)
Bitmap File Visual C# Items
Class Diagram Visual C# Items
Code Analysis Rule Set Visual C# Items
Code File Visual C# Items
CoffeeScript File Visual C# Items
Cursor File Visual C# Items
Custom Control Visual C# Items
DataSet Visual C# Items
Debugger Visualizer Visual C# Items
Domain Service Class Visual C# Items
Dynamic Data Field Visual C# Items
EF 5.x DbContext Generator Visual C# Items
HTML Page Visual C# Items
Type: Visual C# Items
A DataSet for using data in your application
Name: DataSet1.xsd
Add
Cancel

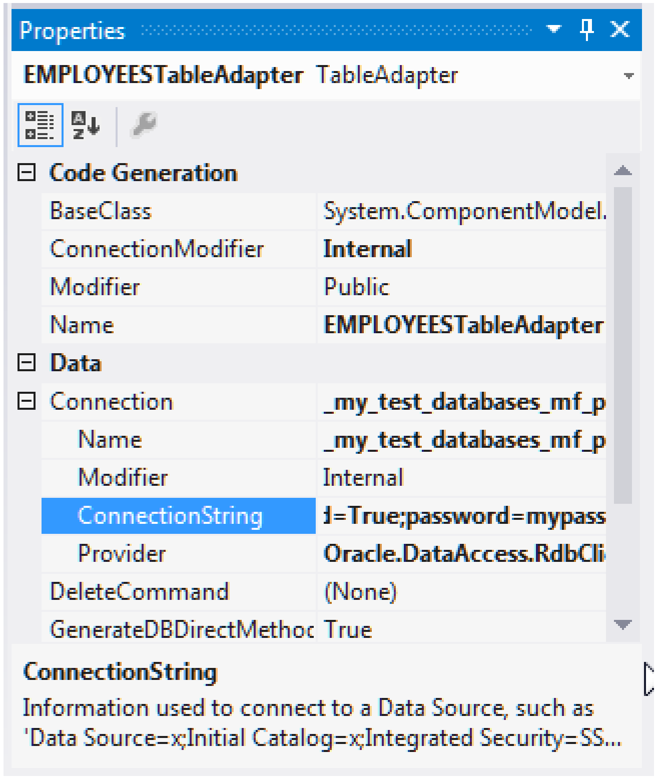
From the Microsoft Server Explorer, drag and drop "Employees" table onto the open designer.



This will create a datasource and an associated **EMPLOYEESTableAdapter** that can you use in your application to access the employees table.



Step3: Ensure the password is added to the connection string. Select **EMPLOYEESTableAdapter**, and right click to view its properties. Click in the ConnectionString field and append your password in the below format **Password=yourpassword**



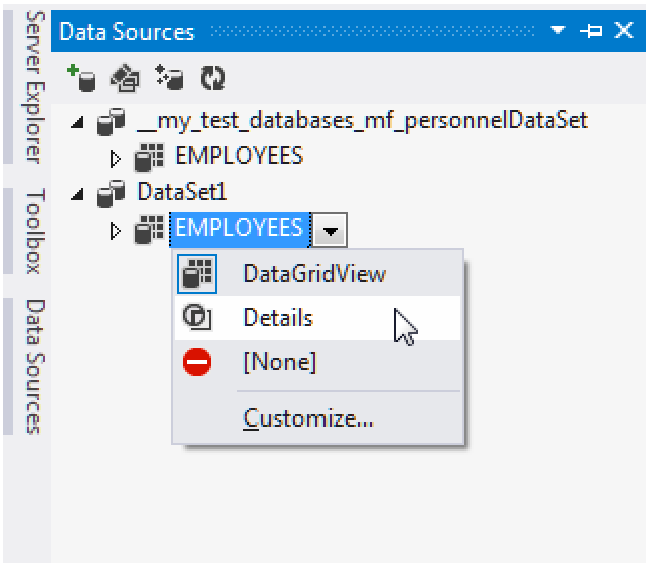
Code Generation	
BaseClass	System.ComponentModel.
ConnectionModifier	**Internal**
Modifier	Public
Name	**EMPLOYEESTableAdapter**
Data	
Connection	**_my_test_databases_mf_p**
Name	**_my_test_databases_mf_p**
Modifier	Internal
ConnectionString	**d=True;password=mypass**
Provider	**Oracle.DataAccess.RdbCli**
DeleteCommand	(None)
GenerateDBDirectMethoc	True

ConnectionString
Information used to connect to a Data Source, such as 'Data Source=x;Initial Catalog=x;Integrated Security=SS...

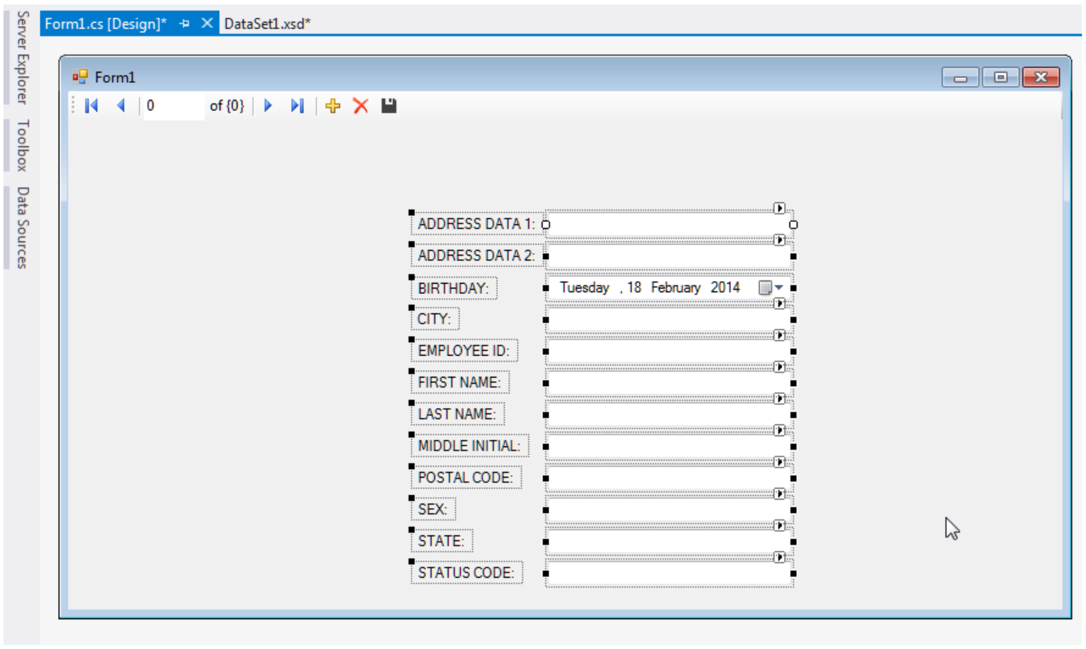
7.4 Designing the windows application

To design the application:

Step1: Ensure the Form1.cs is open in design mode.
Step2: Using the Microsoft data sources window, use the smart menu on the Employees data source, and select **Details**



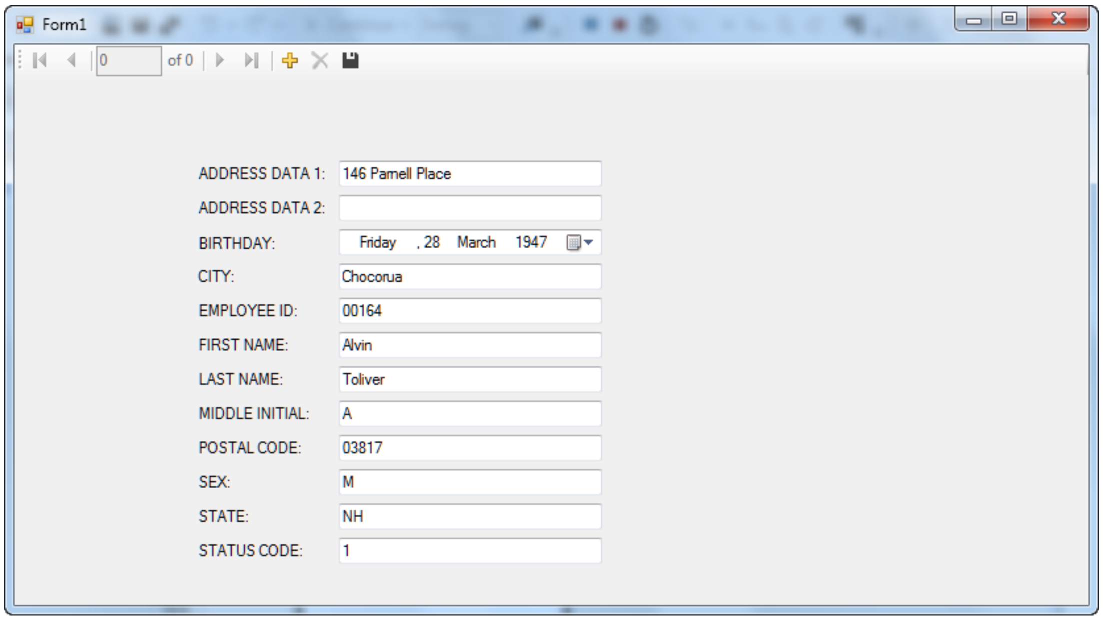
Step3: Drag and drop the Employees data source from the data source window onto the form.



A prototype form to access the employees data will be created for you.

7.5 Running the windows application

Run the application using the top-level menu. You can insert, delete rows and change the existing data.



ADDRESS DATA 1:	146 Parnell Place
ADDRESS DATA 2:	
BIRTHDAY:	Friday , 28 March 1947
CITY:	Chocorua
EMPLOYEE ID:	00164
FIRST NAME:	Alvin
LAST NAME:	Toliver
MIDDLE INITIAL:	A
POSTAL CODE:	03817
SEX:	M
STATE:	NH
STATUS CODE:	1

Show Navigation

Chapter 8 Configuration wizard and generating queries

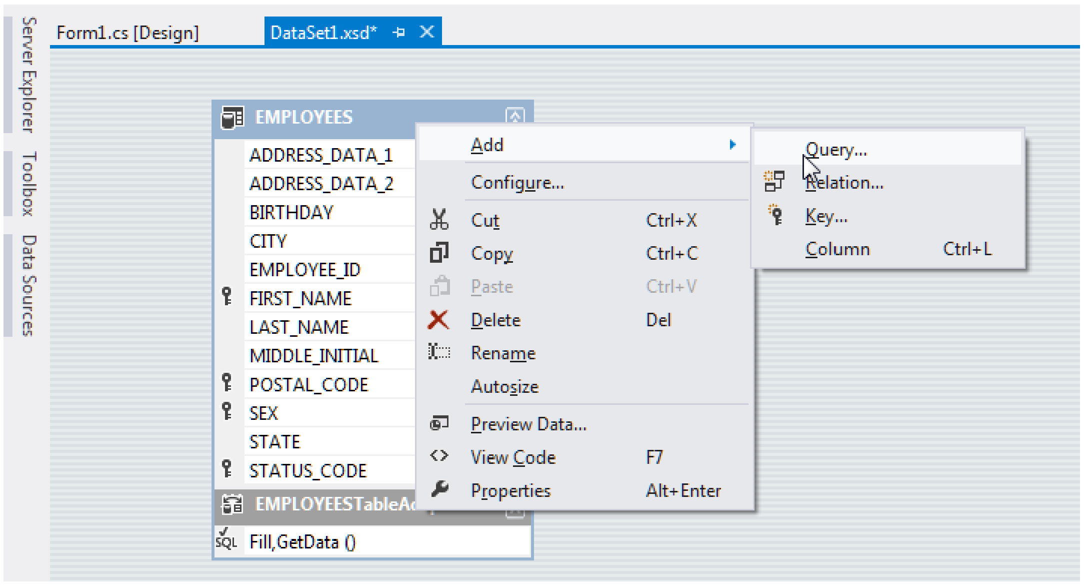
Wizards make it easy to perform many operations within Visual studio. The Table Adapter Query Configuration Wizard simplifies the creation of queries you may execute on your Rdb database.

This chapter contains:

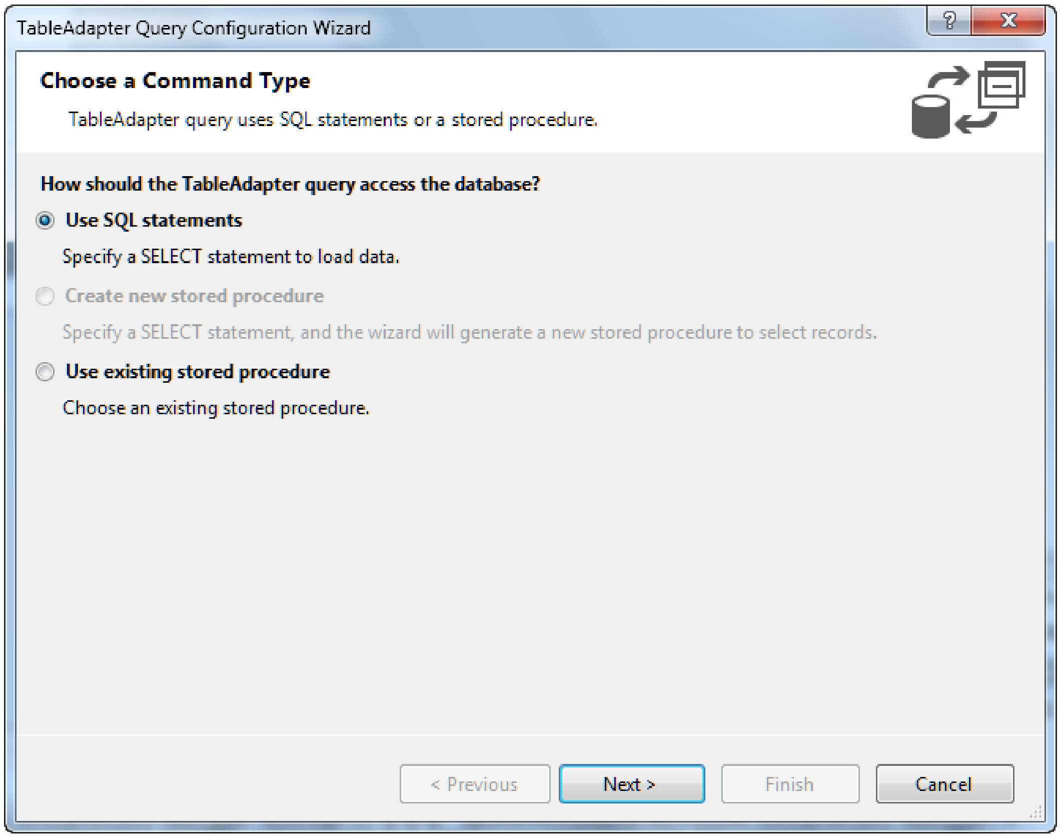
8.1 Using the Query Configuration Wizard

To use the Query Configuration Wizard:

Step1: In the DataSet XSD window right click dataset and click **add→Query**



The Table Adapter Query Configuration Wizard will be shown.



TableAdapter Query Configuration Wizard

Choose a Command Type

TableAdapter query uses SQL statements or a stored procedure.

How should the TableAdapter query access the database?

[✓ **Use SQL statements**]

Specify a SELECT statement to load data.

Create new stored procedure

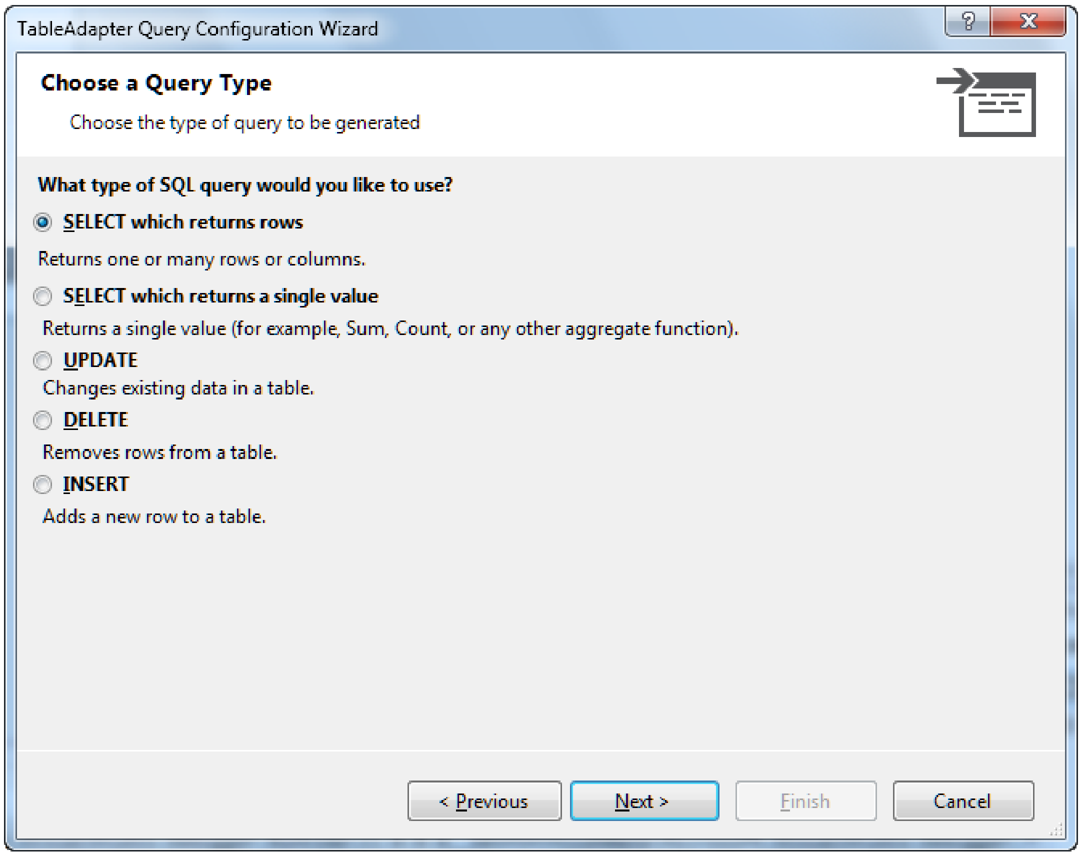
Specify a SELECT statement, and the wizard will generate a new stored procedure to select records.

Use existing stored procedure

Choose an existing stored procedure.

Step2: Select **Use SQL Statement**

Clicking **Next** will bring up the query Type window. This window allows you to choose from several types of standard SQL statements.



TableAdapter Query Configuration Wizard

Choose a Query Type

Choose the type of query to be generated

What type of SQL query would you like to use?

[✓ **SELECT which returns rows**]
Returns one or many rows or columns.

SELECT which returns a single value
Returns a single value (for example, Sum, Count, or any other aggregate function).

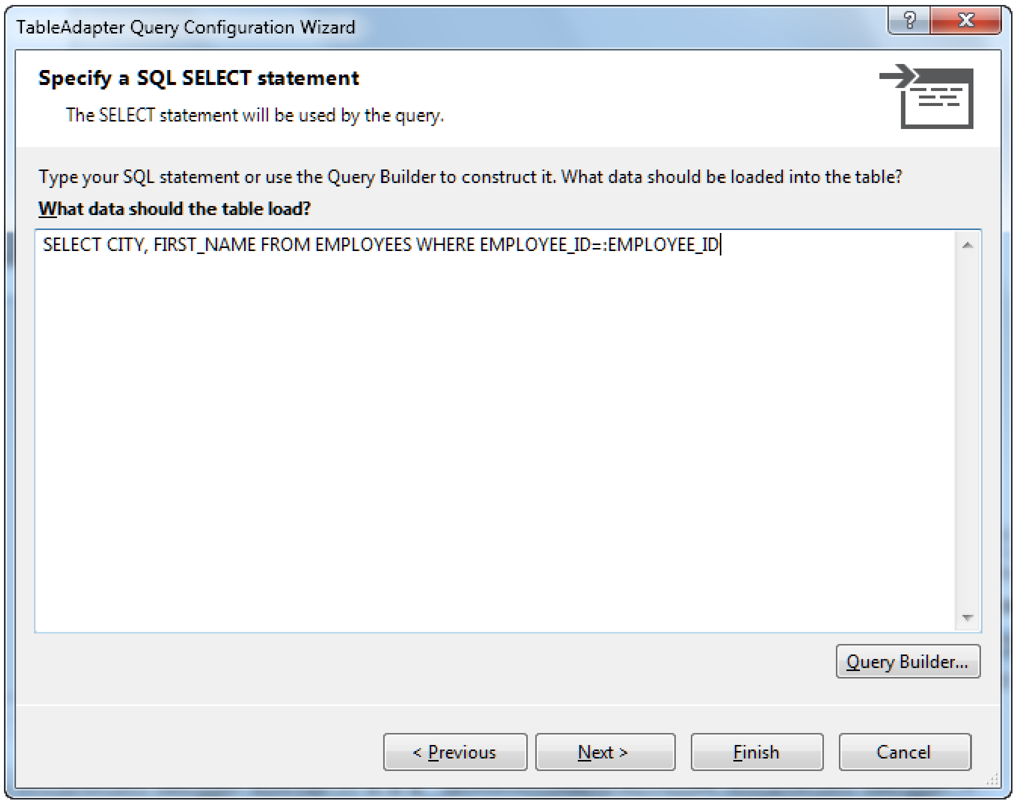
UPDATE
Changes existing data in a table.

DELETE
Removes rows from a table.

INSERT
Adds a new row to a table.

Step3: Select the query type **SELECT which returns rows.**

Step4: Click **Next** to bring up the query specification window. This is where you may enter a SQL query statement.. Use the **":"** character pre-pended to parameter names to specify parameters you wish to use.


TableAdapter Query Configuration Wizard

Specify a SQL SELECT statement

The SELECT statement will be used by the query.

Type your SQL statement or use the Query Builder to construct it. What data should be loaded into the table?

What data should the table load?

```
SELECT CITY, FIRST_NAME FROM EMPLOYEES WHERE EMPLOYEE_ID=:EMPLOYEE_ID
```

For example to return details of an employee giving the **employee_id**:

```
SELECT CITY, FIRST_NAME FROM EMPLOYEES WHERE EMPLOYEE_ID=:EMPLOYEE_ID
```

Step5: Click **Finish**. We have just added a select functionality to the typed dataset using **TableAdapter** without writing a single line of code.

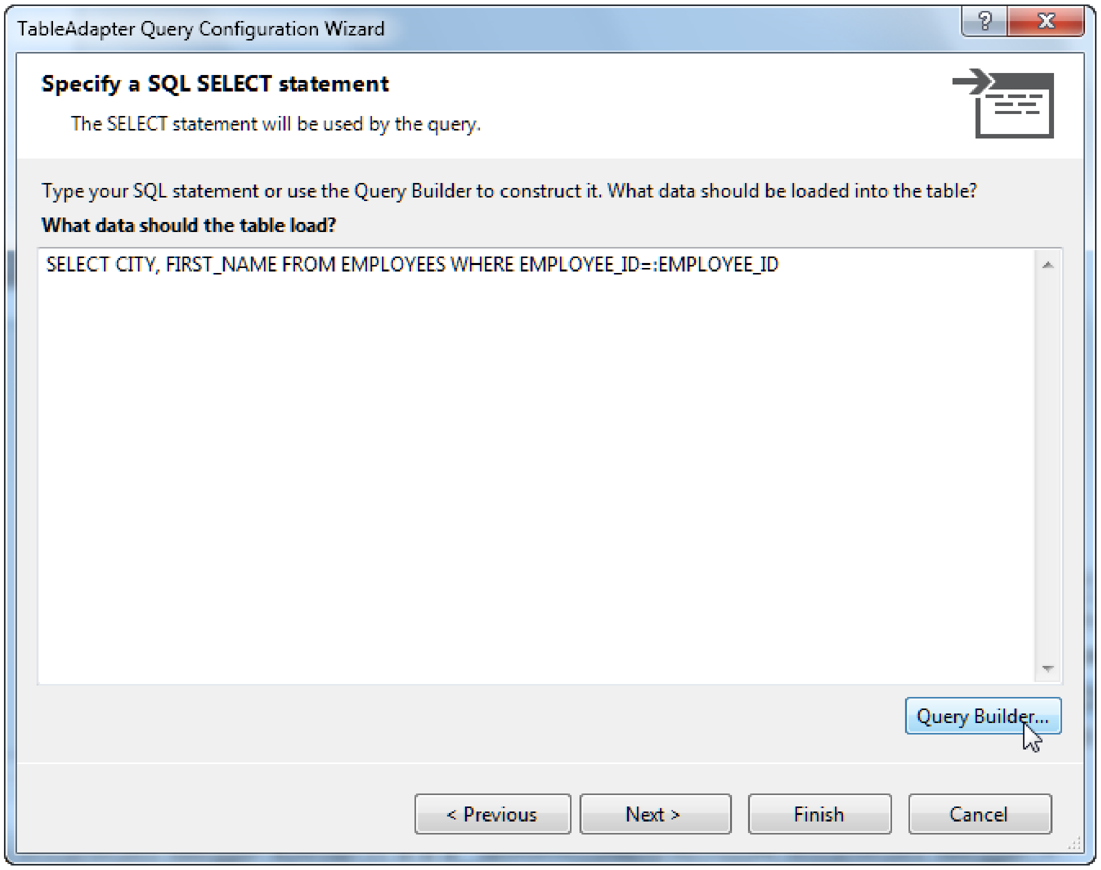
Repeat the above steps to produce other statements such as **UPDATE** and **DELETE** .

8.2 Using the Query Builder

Instead of entering the SQL text directly Visual Studio provides the **Query Builder** option within the configuration Wizard that allows quick and easy creation of syntactically correct SQL queries on your datasets.

Using Query Builder to generate queries:

Step1: Startjng with Query Configuration Wizard used in the previous walk-through, Click **Query Builder** to bring up the builder window .


TableAdapter Query Configuration Wizard
Specify a SQL SELECT statement
The SELECT statement will be used by the query.

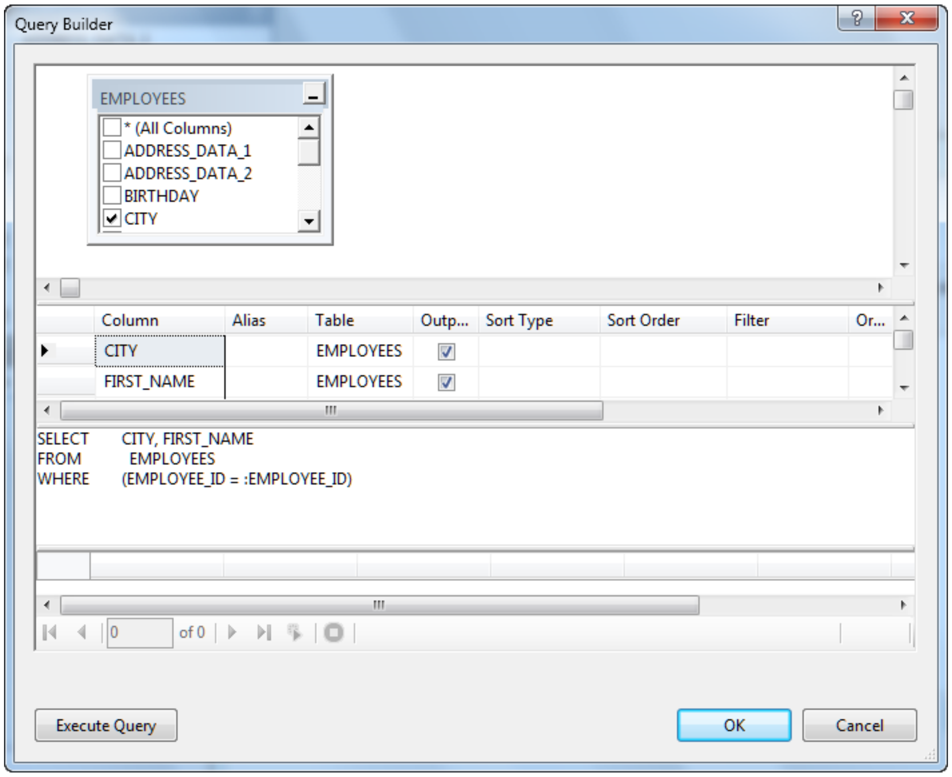
Type your SQL statement or use the Query Builder to construct it. What data should be loaded into the table?

What data should the table load?

```
SELECT CITY, FIRST_NAME FROM EMPLOYEES WHERE EMPLOYEE_ID=:EMPLOYEE_ID
```

Query Builder...
< Previous
Next >
Finish
Cancel

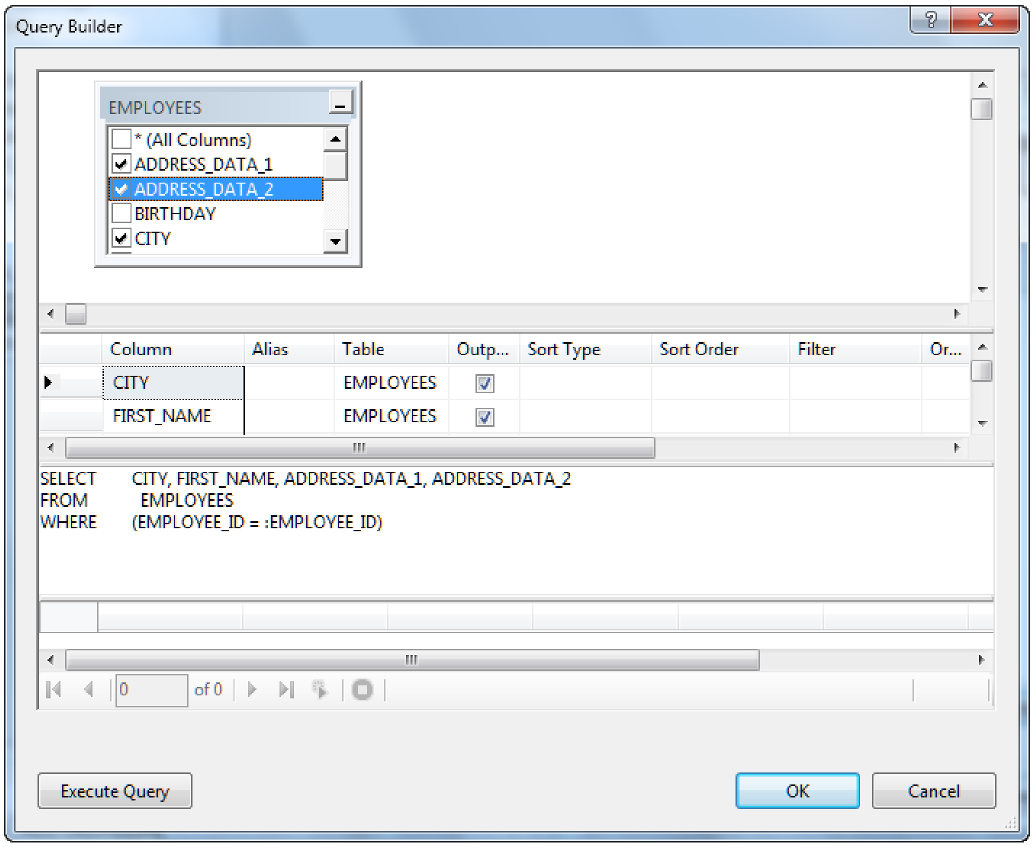
Step2: This will bring up a window allowing you to choose columns for your query, in this case it will be preloaded with the query we typed in before.



Column	Alias	Table	Outp...	Sort Type	Sort Order	Filter	Or...
CITY		EMPLOYEES	✓				
FIRST_NAME		EMPLOYEES	✓				

```
SELECT     CITY, FIRST_NAME
FROM        EMPLOYEES
WHERE     (EMPLOYEE_ID = :EMPLOYEE_ID)
```

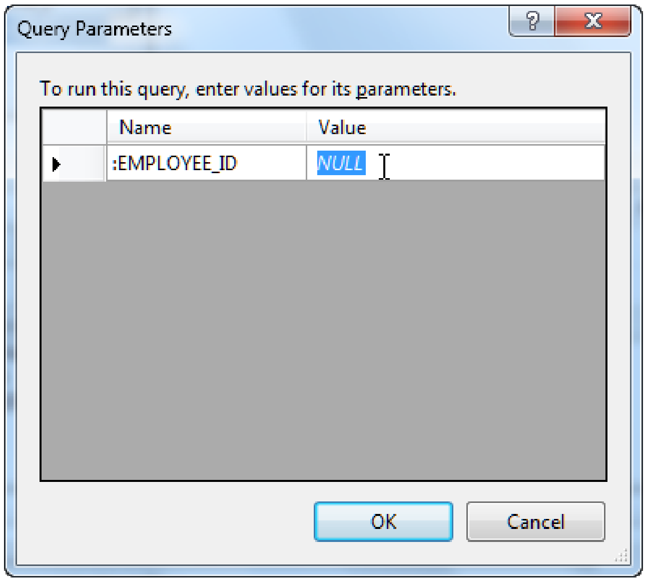
Step3: Within the **EMPLOYEES** table check the columns name and Parameters to the query (Filter) as shown below.



Column	Alias	Table	Outp...	Sort Type	Sort Order	Filter	Or...
CITY		EMPLOYEES	✓				
FIRST_NAME		EMPLOYEES	✓				

```
SELECT     CITY, FIRST_NAME, ADDRESS_DATA_1, ADDRESS_DATA_2
FROM        EMPLOYEES
WHERE      (EMPLOYEE_ID = :EMPLOYEE_ID)
```

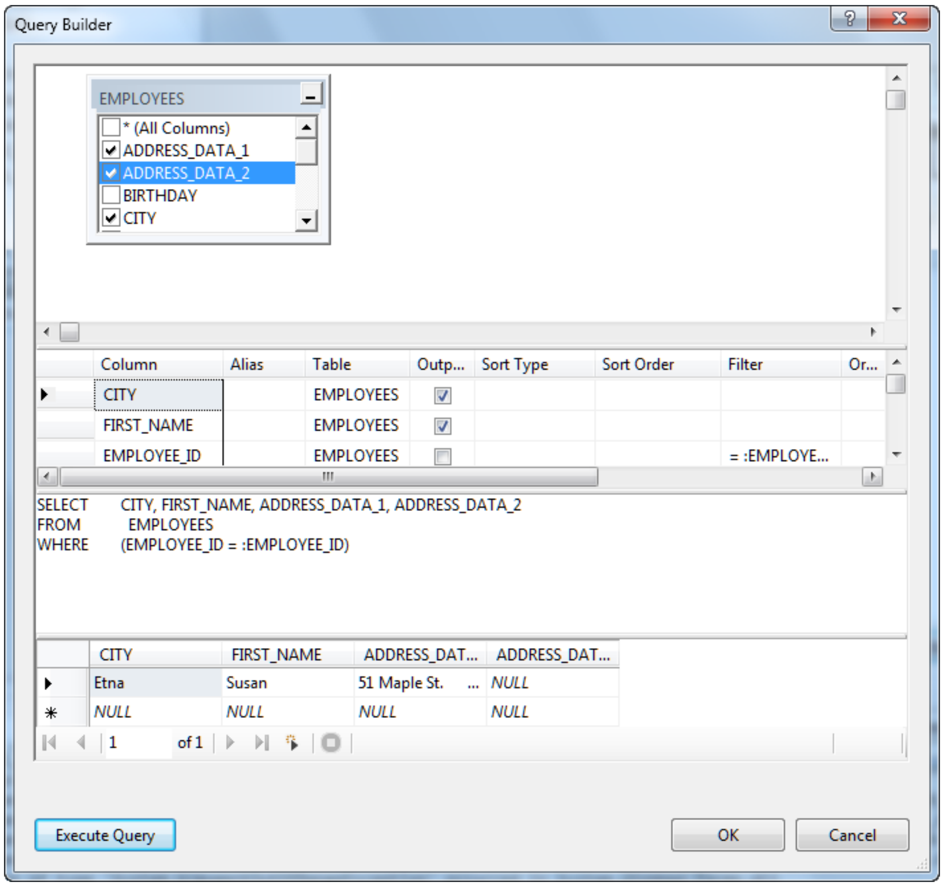
Step4: Click **Execute Query** to execute the generated query. You will be prompted for the employee_id. Enter a valid id and press **OK**.



To run this query, enter values for its parameters.

Name	Value
:EMPLOYEE_ID	NULL

Step5: The results of the operation will be displayed in the results window.



Query Builder

EMPLOYEES

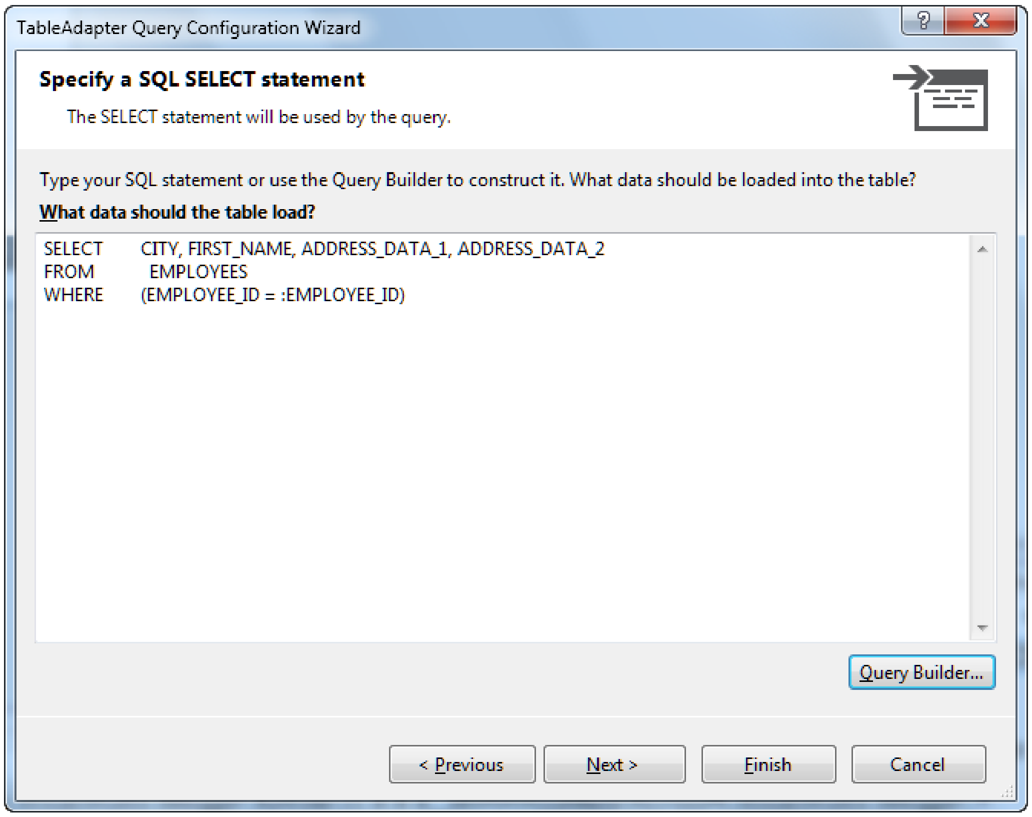
- [ ] * (All Columns)
- [x] ADDRESS_DATA_1
- [x] ADDRESS_DATA_2
- [ ] BIRTHDAY
- [x] CITY

	Column	Alias	Table	Outp...	Sort Type	Sort Order	Filter	Or...
▶	CITY		EMPLOYEES	✓				
	FIRST_NAME		EMPLOYEES	✓				
	EMPLOYEE_ID		EMPLOYEES	☐			= :EMPLOYE...	

```
SELECT      CITY, FIRST_NAME, ADDRESS_DATA_1, ADDRESS_DATA_2
FROM          EMPLOYEES
WHERE       (EMPLOYEE_ID = :EMPLOYEE_ID)
```

	CITY	FIRST_NAME	ADDRESS_DAT...	ADDRESS_DAT...
▶	Etna	Susan	51 Maple St. ...	*NULL*
*	*NULL*	*NULL*	*NULL*	*NULL*

Step6: Click "**OK**" to accept the query .


TableAdapter Query Configuration Wizard
Specify a SQL SELECT statement
The SELECT statement will be used by the query.
Type your SQL statement or use the Query Builder to construct it. What data should be loaded into the table?
What data should the table load?
Query Builder...
< Previous
Next >
Finish
Cancel

```
SELECT     CITY, FIRST_NAME, ADDRESS_DATA_1, ADDRESS_DATA_2
FROM        EMPLOYEES
WHERE      (EMPLOYEE_ID = :EMPLOYEE_ID)
```

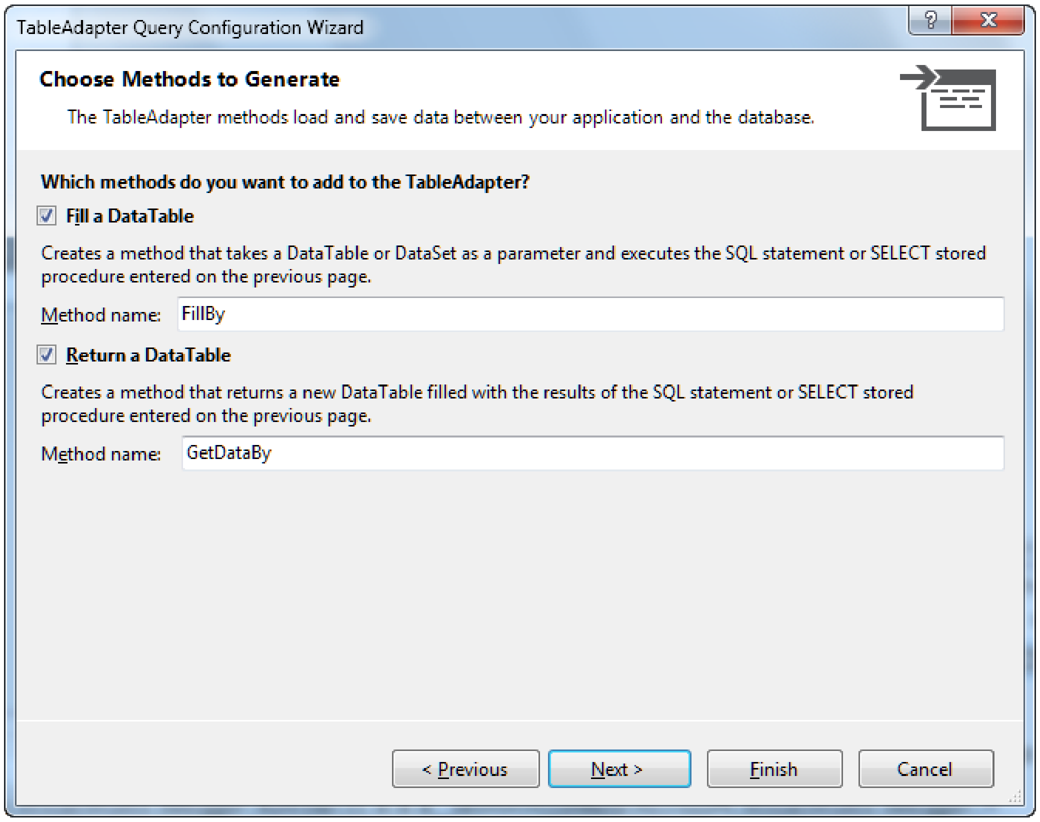
Step7: Click "**Finish**". Your query is now ready to use. The Query Configuration Wizard will now display the Methods Generation window.

8.3 Methods Generation

Once you have generated a query, this query may be used within the TableAdapter . The next step is to choose what methods should be available within your TableAdapter.

How to choose methods:

Step1: Press NEXT in the Query Configuration Wizard to bring up the choose Methods To Generate panel.


TableAdapter Query Configuration Wizard

Choose Methods to Generate

The TableAdapter methods load and save data between your application and the database.

Which methods do you want to add to the TableAdapter?

☑ **Fill a DataTable**

Creates a method that takes a DataTable or DataSet as a parameter and executes the SQL statement or SELECT stored procedure entered on the previous page.

Method name: FillBy

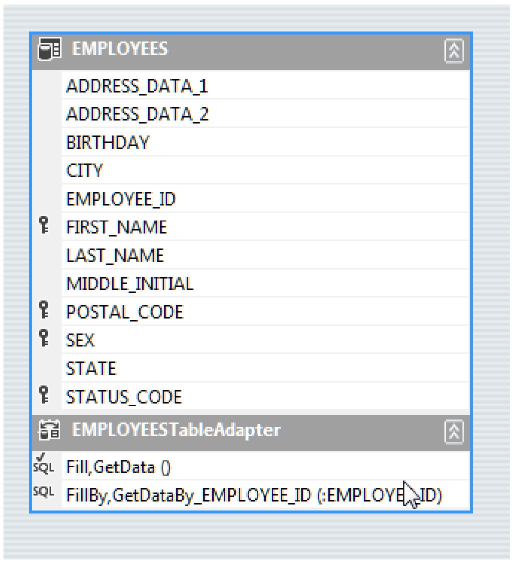
☑ **Return a DataTable**

Creates a method that returns a new DataTable filled with the results of the SQL statement or SELECT stored procedure entered on the previous page.

Method name: GetDataBy

Step2: Select the **Fill A DataTable** and **Return a DataTable** .
Step3: Change the **Return a DataTable** method name to
GetDataBy_EMPID
Step4: Press **FINISH.**

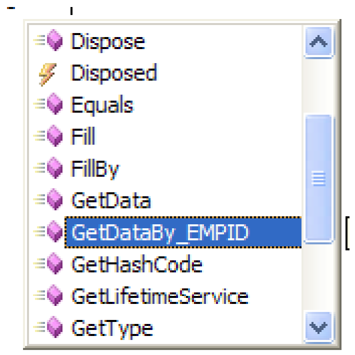
The final dataset in the designer show look like the following:


EMPLOYEES
ADDRESS_DATA_1
ADDRESS_DATA_2
BIRTHDAY
CITY
EMPLOYEE_ID
FIRST_NAME
LAST_NAME
MIDDLE_INITIAL
POSTAL_CODE
SEX
STATE
STATUS_CODE
EMPLOYEESTableAdapter
Fill,GetData ()
FillBy,GetDataBy_EMPLOYEE_ID (:EMPLOYEE_ID)

Step5: To execute the above functionality in your code, instantiate the table adapter as follows:

```
DataSet1TableAdapters.EMPLOYEESTableAdapter adapt = new
DataSet1TableAdapters.EMPLOYEESTableAdapter();
```

This new select function can now also be seen in the TableAdapter intellisense.



Step6: The methods provide by the TableAdapter may be used within your code

```
// Get Employee details by employee_id

dataGridView1.DataSource=adapt.GetDataBy_EMPID(00164);
```

Show Navigation

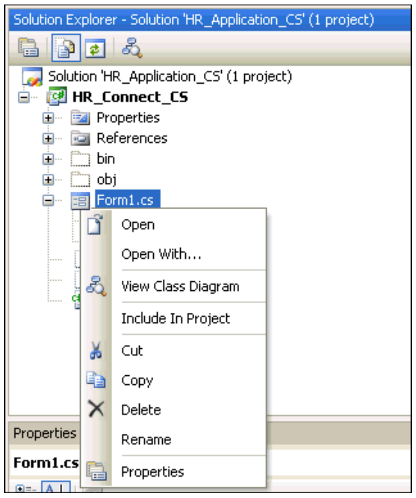
Chapter 9 Copying a Form

Because you will be using this application to learn about various aspects of application development with Oracle Rdb, you should make copies of your form for reuse.

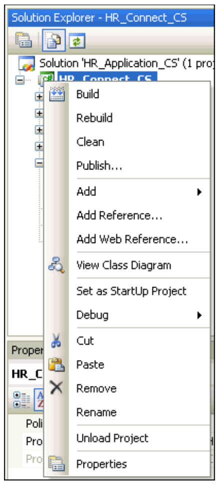
To create a copy of an existing form:

Step1: In the Solution Explorer, right-click on `Form1.xx` or any other file you need to copy. Select **Copy**.

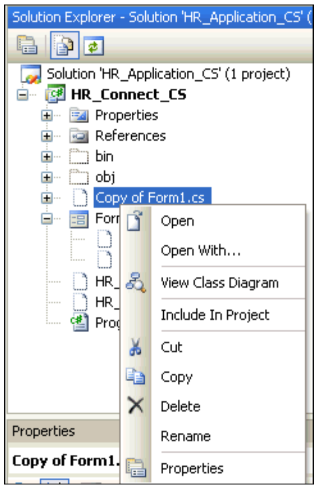
If `Form1.xx` does not appear in the Solution Explorer, from the Project menu, select **Show All Files**.



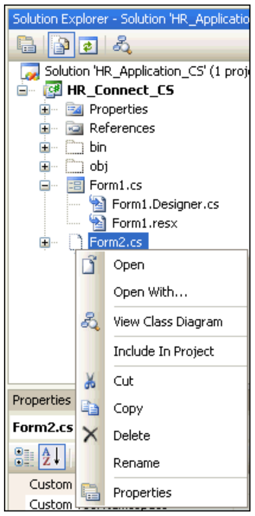
Step2: Right-click **HR_Connect_CS** or other project. Select **Paste**.



Step3: Right-click **Copy of `Form1.cs`**. Select **Rename**. Change the name of the form to `Form2.cs`.



Step4: Right-click on `Form1.cs`, and select **Include In Project**.



Step5: Right-click on `Form1.cs`, and select **Exclude From Project**.

You can include and exclude forms from the project just by reversing these steps.

Note:

This process generally works smoothly. If you encounter a problem, try running Rebuild Solution from the Build menu.

Show Navigation

Chapter 10 Oracle Rdb Entity Framework Provider

The Entity Framework is a new part of ADO.NET that allows you to build your applications against conceptual data models. It provides a greater level of abstraction and supports code that is independent of any particular relational database.

It provides an Entity Data Model (EDM) for defining data at the database and conceptual level and mapping between the two. It includes a nice set of tools that can be used to generate the EDM and corresponding objects that represent the database. This eliminates much of the required boilerplate data access code, and makes it a snap to create data-centric applications.

The chapter also provides some simple examples on how to carry out Model-First and Database-First application development using the Oracle Rdb Entity Framework Provider.

Model-first is the ability to start with a conceptual model and create the database from it. Database-first is the ability to start with the database and build your EDM from the database objects.

The examples in this chapter are based on Visual Studio 2010. If you are using another version of Visual Studio, the user interface may have changed. Please refer to your Visual studio documentation to see how the same objectives may be achieved.

This chapter contains:

10.1 Creating new Windows Form Application

The first step is to create a new Windows Forms application:

Step1: In Visual Studio IDE, select **File→New→Project** from the main menu.

Choose the Windows Forms Application installed template. Click **OK.** The solution is created.

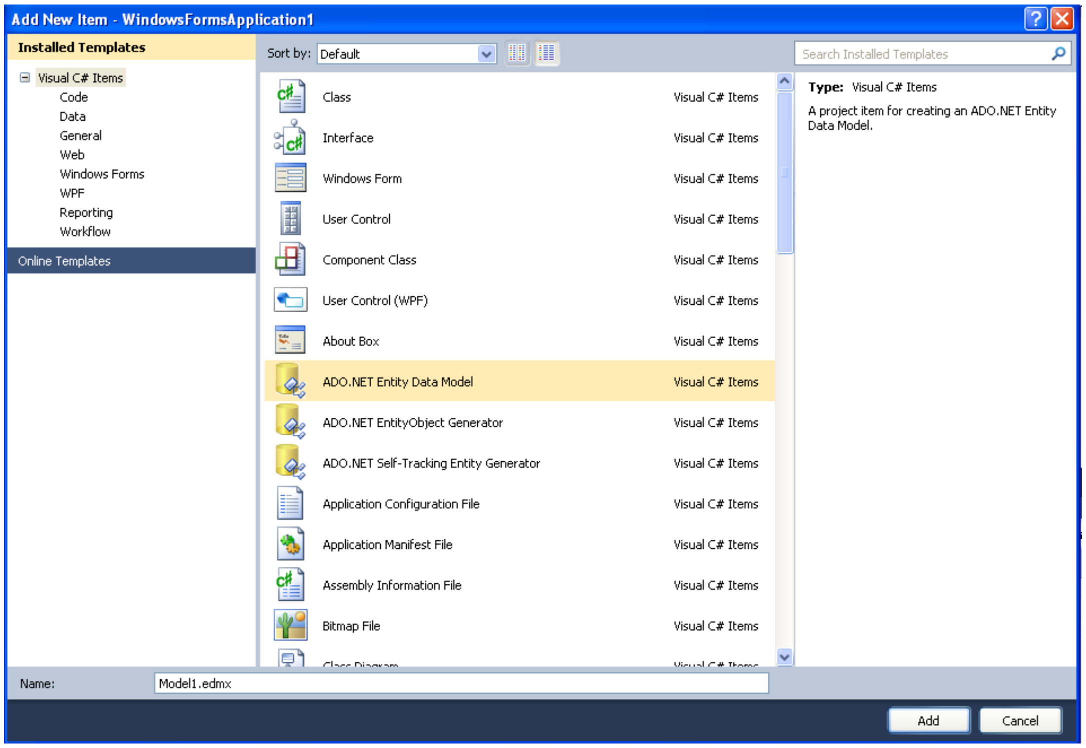
Step2: Right-click on `Form1.cs` and select **Exclude From Project**.

You can now proceed to generate a Entity Data Model from an existing database schema.

10.1.1 Adding an Entity Data Model

How to add an Entity Data Model:

Step1: In the Solution Explorer, right-click your application and select **Add→ New Item.** This opens **Visual Studio installed templates** screen.
Step2: From **Visual Studio installed templates**, select **ADO.NET Entity Data Model**. Click **Add**.


Add New Item - WindowsFormsApplication1
Installed Templates
Visual C# Items
Code
Data
General
Web
Windows Forms
WPF
Reporting
Workflow
Online Templates
Sort by: Default
Search Installed Templates
Class
Interface
Windows Form
User Control
Component Class
User Control (WPF)
About Box
ADO.NET Entity Data Model
ADO.NET EntityObject Generator
ADO.NET Self-Tracking Entity Generator
Application Configuration File
Application Manifest File
Assembly Information File
Bitmap File
Visual C# Items
Add
Cancel

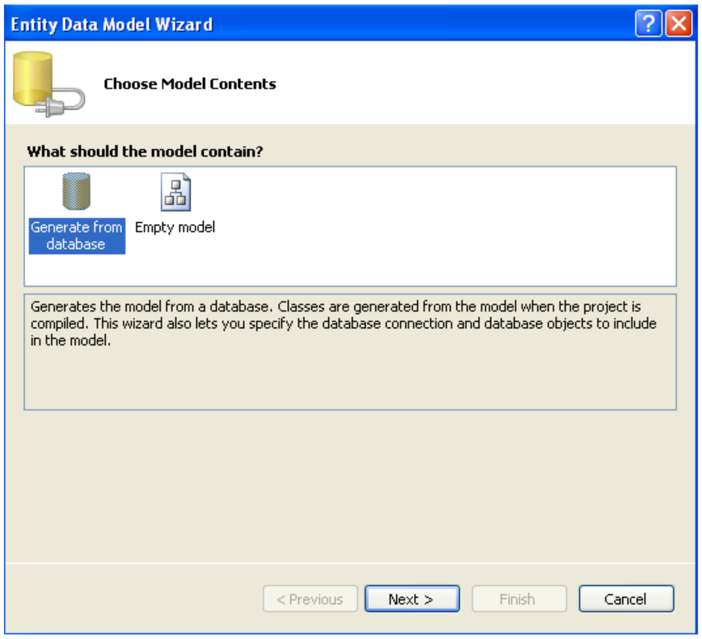
Type: Visual C# Items

A project item for creating an ADO.NET Entity Data Model.

Name: Model1.edmx

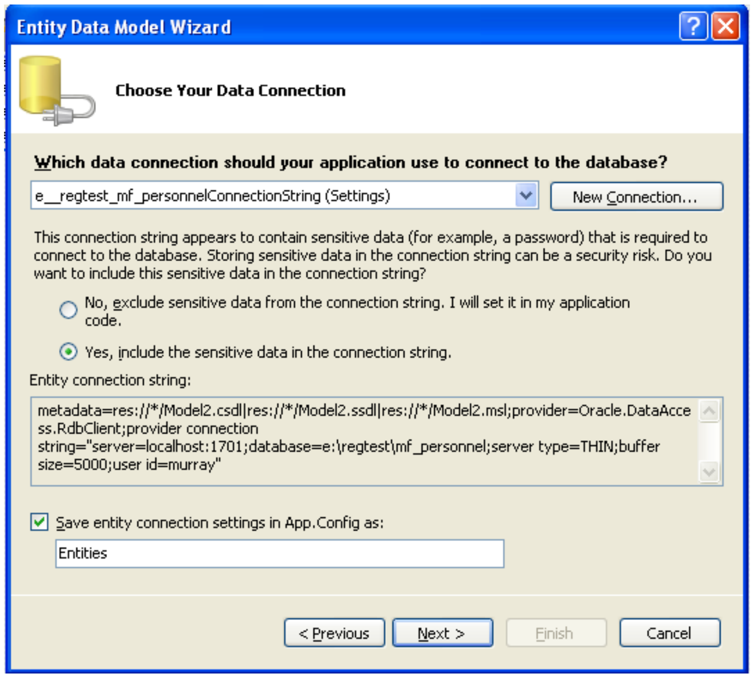
You will now see the **Entity Data Model Wizard**. This wizard is used to generate the Entity Data Model from the database specified in the database connection string (in next step).

Step3: Select the icon **Generate from database**. Click **Next**.



Generates the model from a database. Classes are generated from the model when the project is compiled. This wizard also lets you specify the database connection and database objects to include in the model.

Step4: You can now select the connection if you have made connection to the test database earlier or create a new connection:



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

e__regtest_mf_personnelConnectionString (Settings)

This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. 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 it in my application code.
- [✓ Yes, include the sensitive data in the connection string.]

Entity connection string:

```
metadata=res://*/Model2.csdl|res://*/Model2.ssdl|res://*/Model2.msl;provider=Oracle.DataAcce
ss.RdbClient;provider connection
string="server=localhost:1701;database=e:\regtest\mf_personnel;server type=THIN;buffer
size=5000;user id=murray"
```

[✓] Save entity connection settings in App.Config as:

Entities

If you have not already done so, you can create the new connection at this time by clicking **New Connection**. See Connecting to the Oracle Rdb Database for details on each fields shown in **New Connection** Form.

Step5: Click **Next**

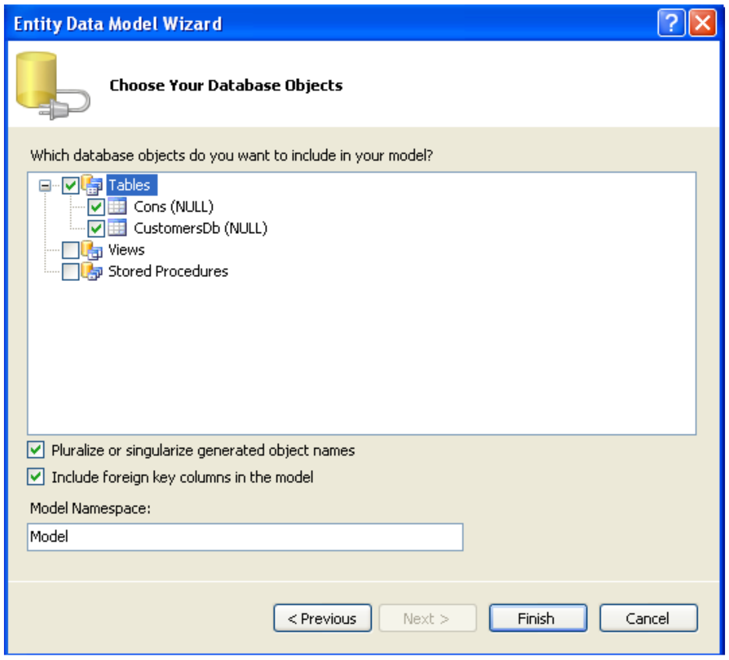
The Entity Data Model Wizard connects to the database. You are then presented with a tree structure of the database.

Here you can select the object you would like to include in your model.

Here we have chosen to select all tables, just as a part of example.

But, You have a option to select any specific table.

Step6: **Check** the box against **Table**s node.



Which database objects do you want to include in your model?

- ✓ Tables
  - ✓ Cons (NULL)
  - ✓ CustomersDb (NULL)
- ☐ Views
- ☐ Stored Procedures

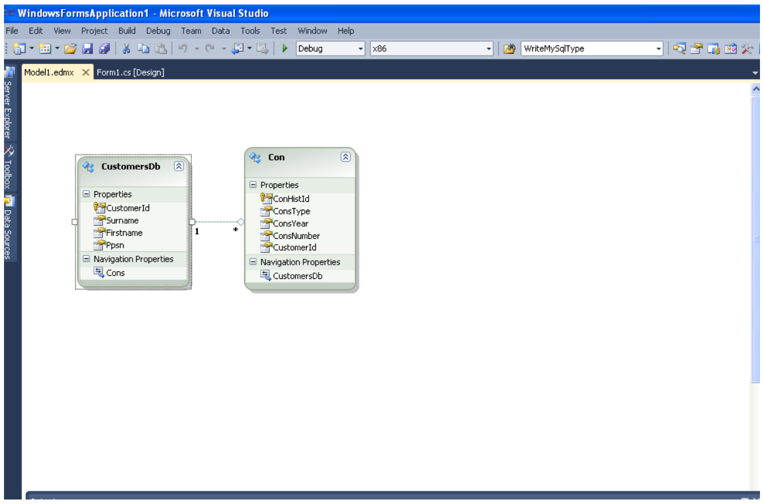
✓ Pluralize or singularize generated object names

✓ Include foreign key columns in the model

Model Namespace: Model

Step7: Click **Finish** to create the model and exit the wizard.

Visual Studio will generate the model then display it.



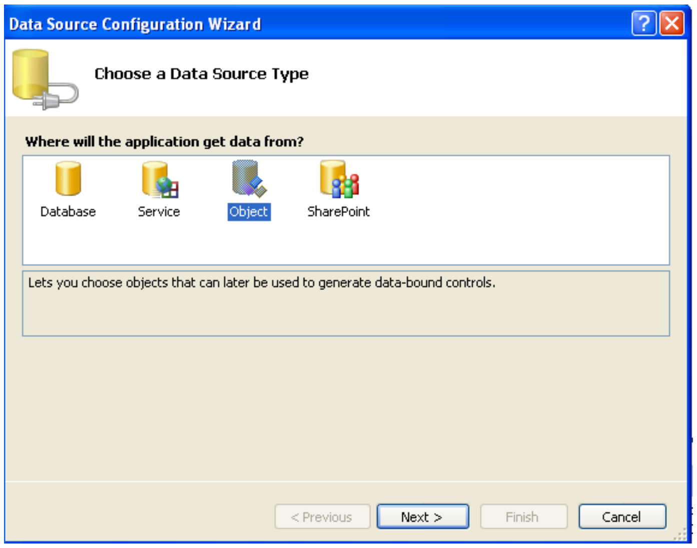
10.1.2 **Adding a new Data Source**

You will now add a new Data Source to your project and see how it can be used to read and write to the database.

To add a new Data source:

Step1: From the Visual Studio Main Menu, select **Data→ Add New Data Source.**

You will be presented with the Data Source Configuration Wizard.



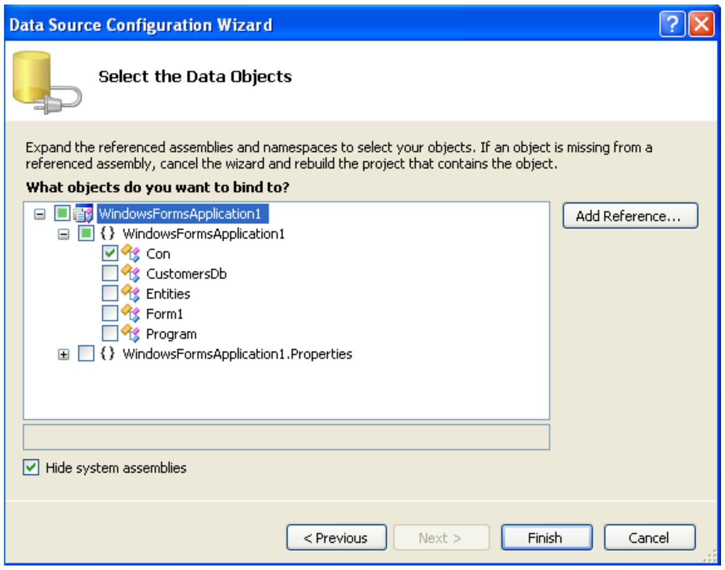
Choose a Data Source Type

Where will the application get data from?

Lets you choose objects that can later be used to generate data-bound controls.

Step2: Select the **Object** icon. Click **Next**.
Step3: You will now select the Object you wish to bind to. Expand the tree.

In this tutorial we have chosen **Con** table as a example. Once the **Con** table has been selected click **Finish**.


Data Source Configuration Wizard

Select the Data Objects

Expand the referenced assemblies and namespaces to select your objects. If an object is missing from a referenced assembly, cancel the wizard and rebuild the project that contains the object.

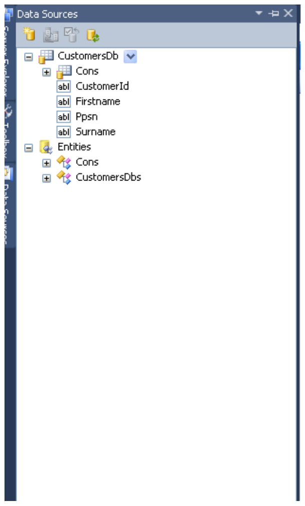
What objects do you want to bind to?

- WindowsFormsApplication1
  - WindowsFormsApplication1
    - [✓] Con
    - [ ] CustomersDb
    - [ ] Entities
    - [ ] Form1
    - [ ] Program
  - [ ] WindowsFormsApplication1.Properties

[✓] Hide system assemblies

The **Con** object will be displayed in the Data Sources panel. If the Data Sources panel is not displayed, select **Data→Show Data** Sources from the Visual Studio Main Menu.

The docked panel will then be displayed.



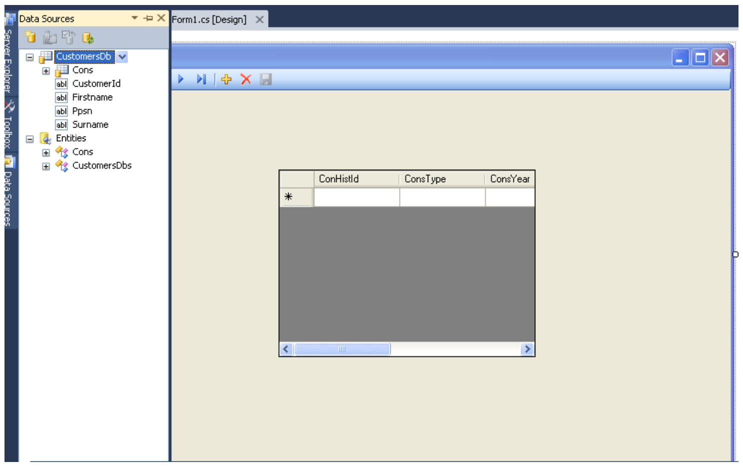
10.1.3 Using the Data Source in a Windows Form

To use the Data Source:

Step1: In the Data Sources panel, select the **Data Source** you just created and **drag and drop** it onto the Form Designer.

By default the Data Source object will be added as a **Data Grid View control**.

Note: the Data Grid View control is bound to the **consBindingSource** and the Navigator control is bound to **consBindingNavigator.**



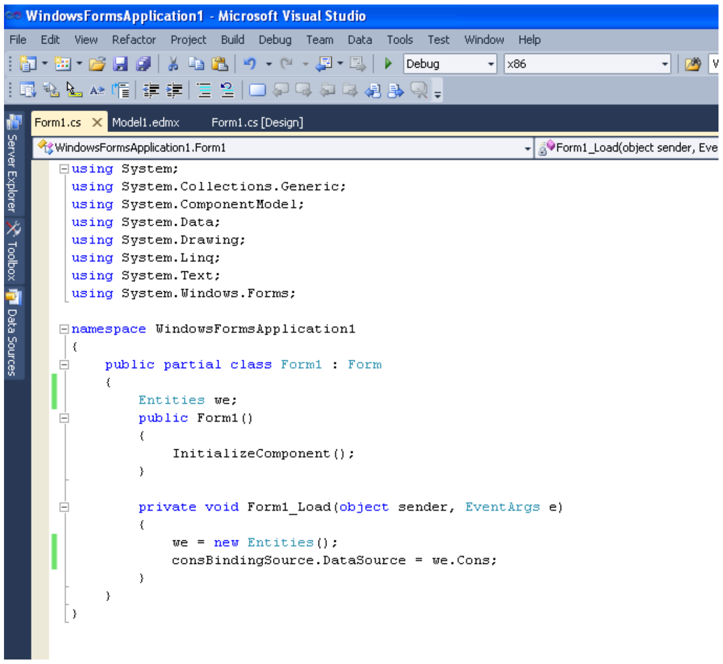
Step2: **Save** and **rebuild** the solution before continuing.

10.1.4 Adding Code to Populate the Data Grid View

You are now ready to add code to ensure that the Data Grid View control will be populated with data from the **Con** database table.

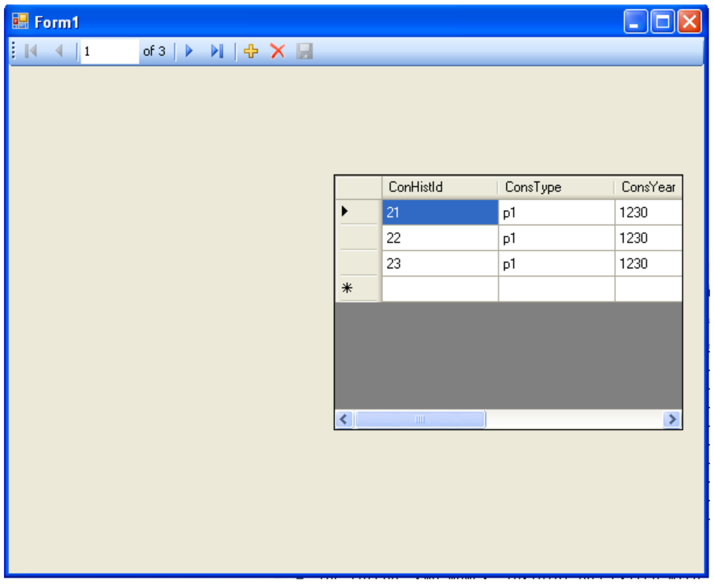
How to add code to Populate the view:

Step1: Double-click the form to access its code.
Step2: Add code to instantiate the Entity Data Model's EntityContainer object and retrieve data from the database to populate the control.


WindowsFormsApplication1 - Microsoft Visual Studio
File Edit View Refactor Project Build Debug Team Data Tools Test Window Help
Debug
x86
Form1.cs
Model1.edmx
Form1.cs [Design]
WindowsFormsApplication1.Form1
Form1_Load(object sender, Eve
Server Explorer
Toolbox
Data Sources

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        Entities we;
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            we = new Entities();
            consBindingSource.DataSource = we.Cons;
        }
    }
}
```

Step3: Save and **rebuild** the solution**.**
Step4: Run the solution. Ensure the grid is populated and you can navigate through the data.



	ConHistId	ConsType	ConsYear
▶	21	p1	1230
	22	p1	1230
	23	p1	1230
*			

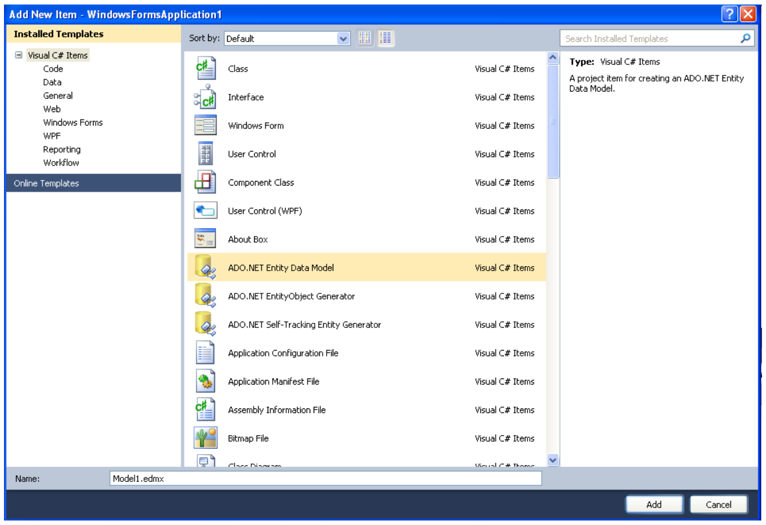
10.2 Create a new Entity

A new Entity Data model can be created from scratch by using the **Entity Data Model Designer.**

How to create an Entity Data Model:

Step1: In the Solution Explorer, right-click your application and select **Add→ New Item.**

Step2: From Visual Studio installed templates, select **ADO.NET Entity Data Model**. Click **Add**.



Add New Item - WindowsFormsApplication1

Installed Templates

- Visual C# Items
  - Code
  - Data
  - General
  - Web
  - Windows Forms
  - WPF
  - Reporting
  - Workflow

Online Templates

Sort by: Default

Search Installed Templates

Class	Visual C# Items
Interface	Visual C# Items
Windows Form	Visual C# Items
User Control	Visual C# Items
Component Class	Visual C# Items
User Control (WPF)	Visual C# Items
About Box	Visual C# Items
ADO.NET Entity Data Model	Visual C# Items
ADO.NET EntityObject Generator	Visual C# Items
ADO.NET Self-Tracking Entity Generator	Visual C# Items
Application Configuration File	Visual C# Items
Application Manifest File	Visual C# Items
Assembly Information File	Visual C# Items
Bitmap File	Visual C# Items

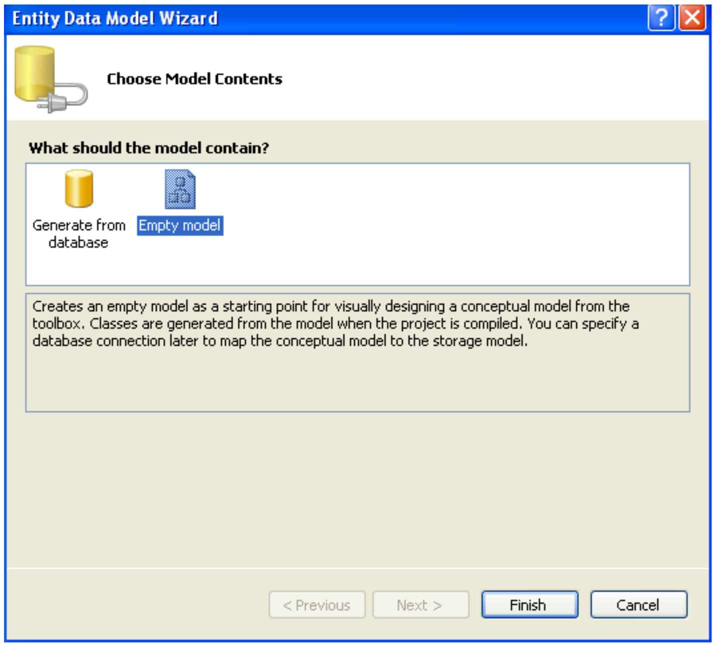
Type: Visual C# Items

A project item for creating an ADO.NET Entity Data Model.

Name: Model1.edmx

You will now see the **Entity Data Model Wizard**. This wizard is used to generate the Entity Data Model from the database specified in the database connection string

Step3: Select the icon **Entity model**. Click **Finish**.



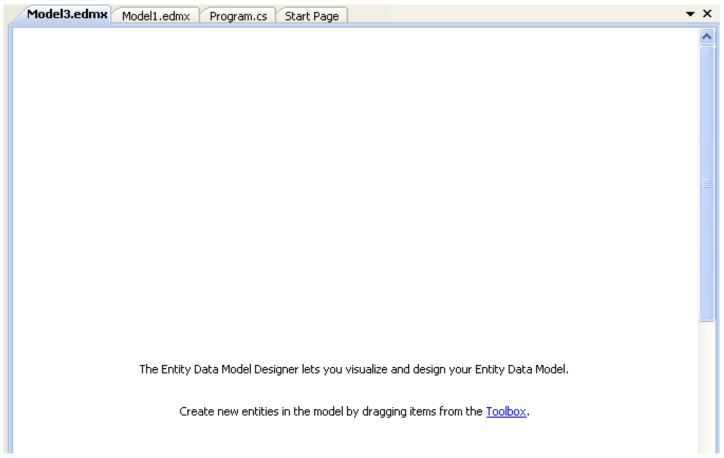
Entity Data Model Wizard

Choose Model Contents

What should the model contain?

Creates an empty model as a starting point for visually designing a conceptual model from the toolbox. Classes are generated from the model when the project is compiled. You can specify a database connection later to map the conceptual model to the storage model.

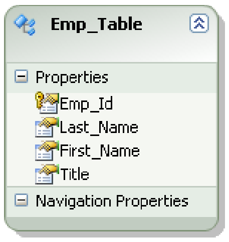
An empty Entity Data Model Designer Form should now be displayed.



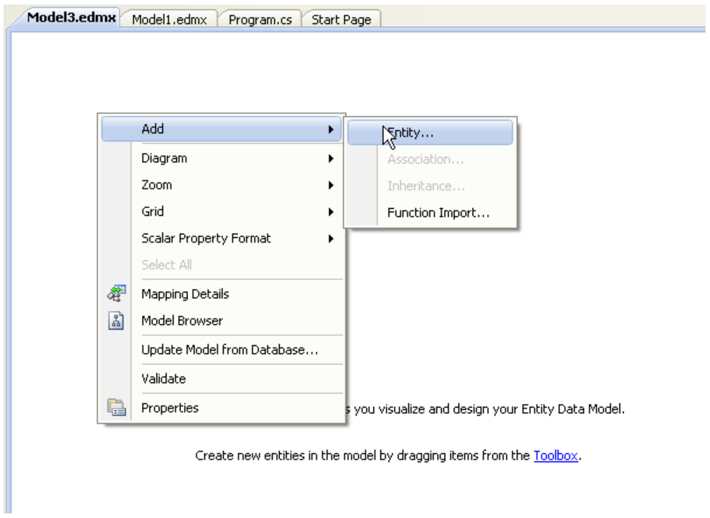
The Entity Data Model Designer lets you visualize and design your Entity Data Model.

Create new entities in the model by dragging items from the Toolbox.

We will now show how to design a following sample entity:



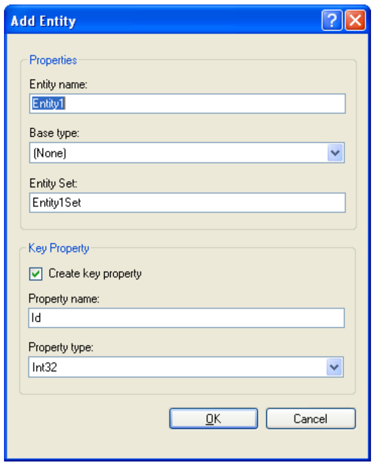
Step4: Add a new Entity by right-clicking in the Model Designer and selecting **Add→ Entity.**



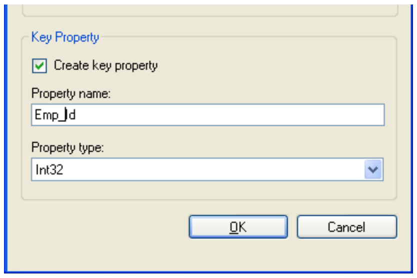
s you visualize and design your Entity Data Model.

Create new entities in the model by dragging items from the Toolbox.

The Add Entity form will be displayed.

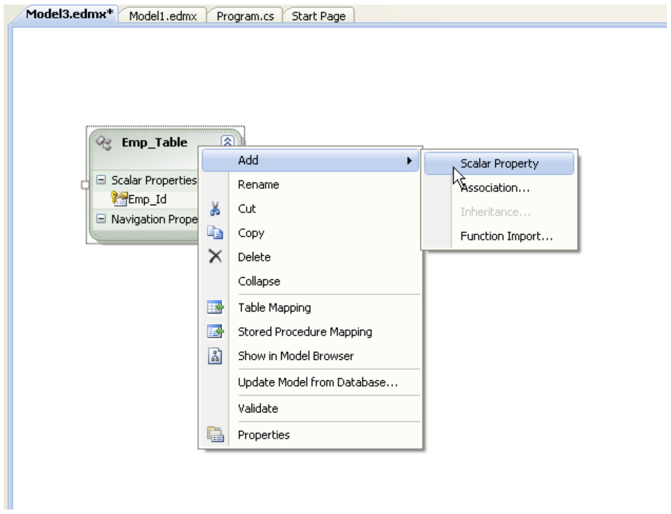


Step5: Change the **Entity name** to `Emp_Table` and the Key **Property name** to `Emp_id`. The Entity Set will be automatically changed to `Emp_TableSet`

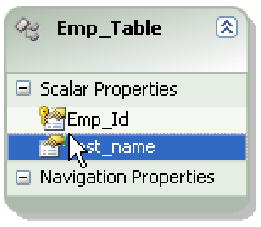


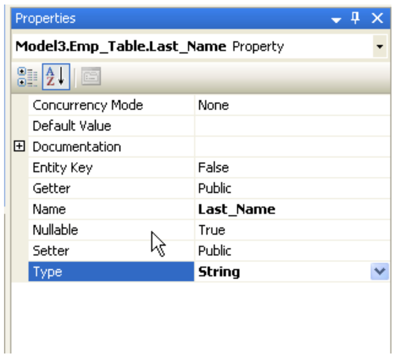
Step6: press **OK**.

Step7: Right-click the new entity and use **Add →Scalar Property** to add three more String properties `Last_name, First_name` and `Title`



Remember to change the **Properties →Type** for each of the new properties to **String** by selecting the respective scalar properties**.**





Model3.Emp_Table.Last_Name Property

Concurrency Mode	None
Default Value	
Documentation	
Entity Key	False
Getter	Public
Name	**Last_Name**
Nullable	True
Setter	Public
Type	**String**

10.3 Creating Complex Types

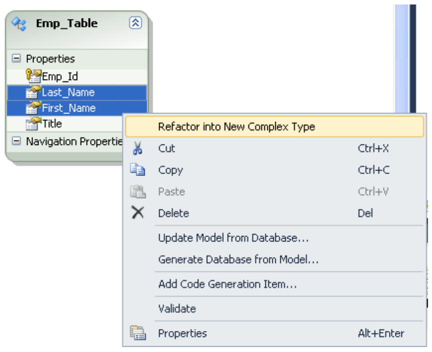
Complex types are non-scalar properties of entity types which enable scalar properties to be organized within entities. This makes easier to work with objects, allowing the grouping of related properties in Entities.

As an example, let us create the Complex type `Employee_Name` instead of the `Last_Name and First_Name` properties.

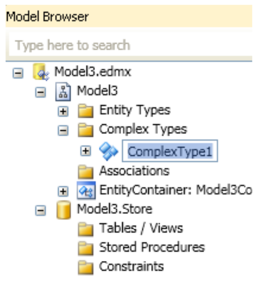
To create a Complex type:

Step1: Hold the **CRTL** key and **Select** the properties in the data model that you want to group together

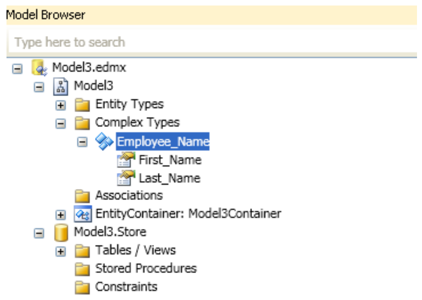
Step2: Right click in the **context menu,** select **Refactor into New Complex Type**



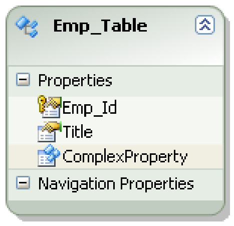
The model browser will display a new Complex Type.



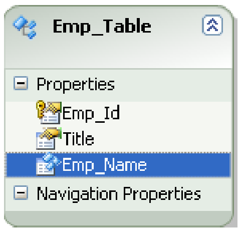
Step3: Now let's give a more meaningful name. If you expand the **Complex Type1**, you will see the properties that we previously selected. Change the name from `ComplexType1` to `Employee_Name`.



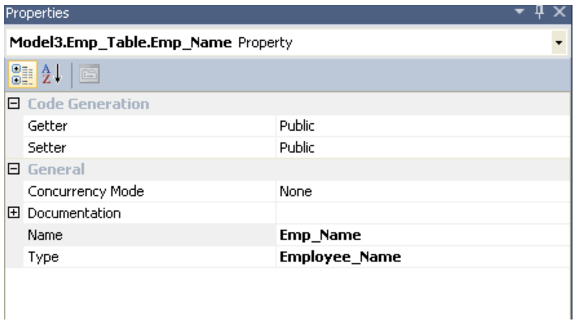
If you look at your data model you will see that **ComplexProperty** has been added to the list.



Step4: Change the name from `ComplexProperty` to `Emp_Name`.



If you look at the properties, you will see that the type of **Emp_Name** is the complex type name that is **Employee_Name** which we just created.



Model3.Emp_Table.Emp_Name Property

Code Generation	
Getter	Public
Setter	Public
General	
Concurrency Mode	None
Documentation	
Name	Emp_Name
Type	Employee_Name

10.3.1 Using the Complex Type created

The Complex type you created may be used as follows:

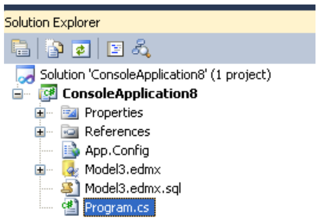
Step1: In Visual Studio, select **File→New→Project** from the main menu.

Choose the Console Application installed template. Click **OK**. The solution is created.

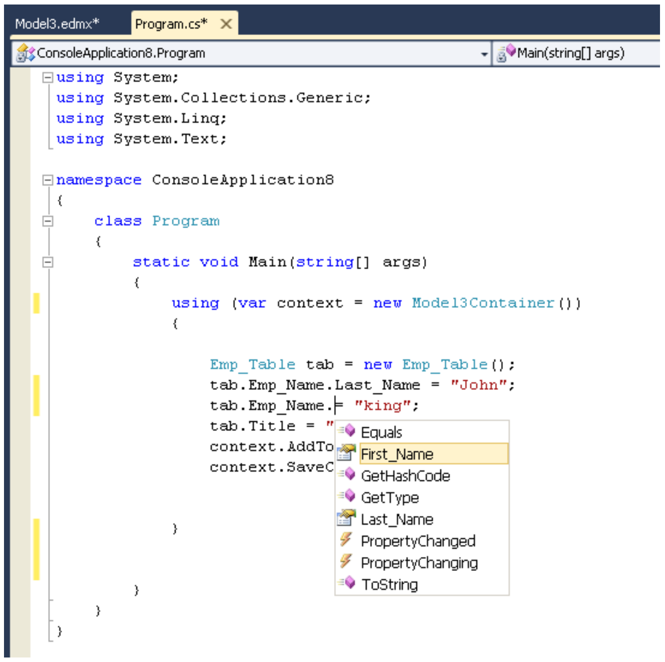
Step2: Design a new Entity with Complex Types. See Create a new Entity , Creating Complex Types

Step3: Generate the Database schema from the data model you have designed. See Generating Database Schema from Model

The Console Application's solution explorer looks as follows:

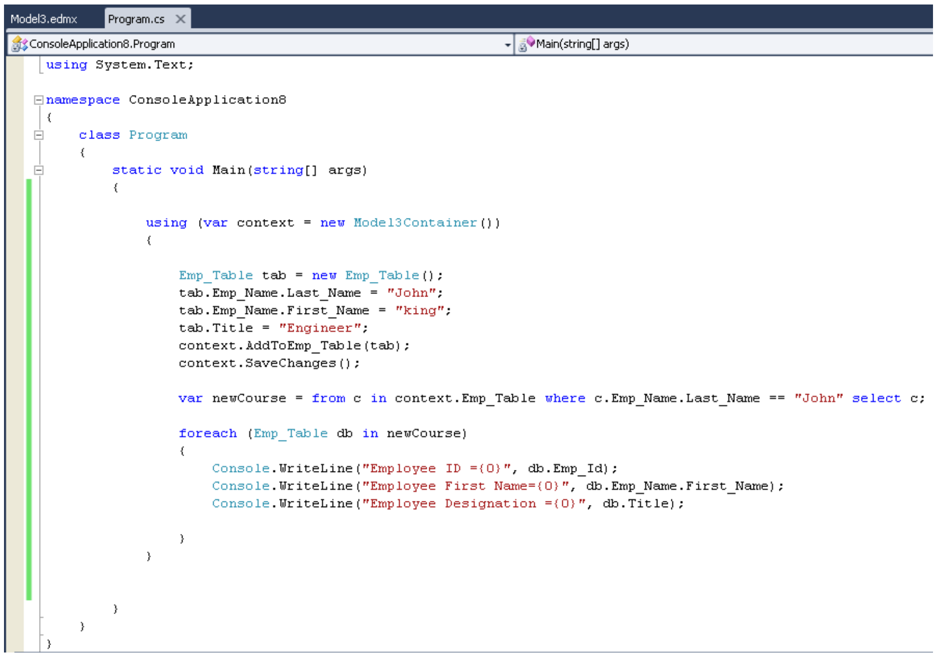


Step4: Open `Program.cs` file by double clicking on it.
Step5: Write the code to use the complex type just created.



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ConsoleApplication8
{
    class Program
    {
        static void Main(string[] args)
        {
            using (var context = new Model3Container())
            {

                Emp_Table tab = new Emp_Table();
                tab.Emp_Name.Last_Name = "John";
                tab.Emp_Name.= "king";
                tab.Title = "
                context.AddTo
                context.SaveC


            }

        }
    }
}
```

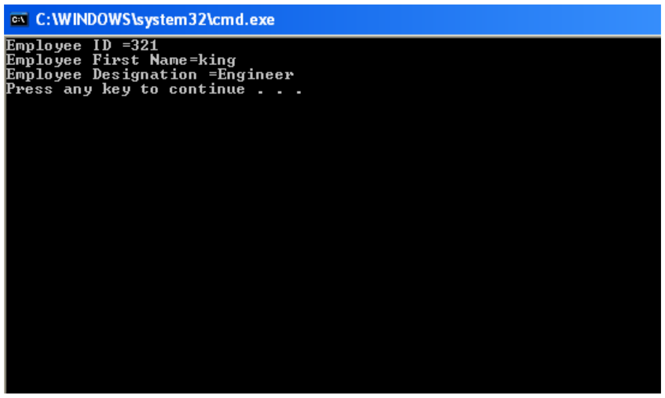
Step6: The Complete code looks as follows.



```
using System.Text;

namespace ConsoleApplication8
{
    class Program
    {
        static void Main(string[] args)
        {

            using (var context = new Model3Container())
            {

                Emp_Table tab = new Emp_Table();
                tab.Emp_Name.Last_Name = "John";
                tab.Emp_Name.First_Name = "king";
                tab.Title = "Engineer";
                context.AddToEmp_Table(tab);
                context.SaveChanges();

                var newCourse = from c in context.Emp_Table where c.Emp_Name.Last_Name == "John" select c;

                foreach (Emp_Table db in newCourse)
                {
                    Console.WriteLine("Employee ID ={0}", db.Emp_Id);
                    Console.WriteLine("Employee First Name={0}", db.Emp_Name.First_Name);
                    Console.WriteLine("Employee Designation ={0}", db.Title);

                }
            }


        }
    }
}
```

Step7: Save and **rebuild** the solution. **See** Compiling and Running the Application

Step8: Run the solution and we get following output on Console.



```
Employee ID =321
Employee First Name=king
Employee Designation =Engineer
Press any key to continue . . .
```

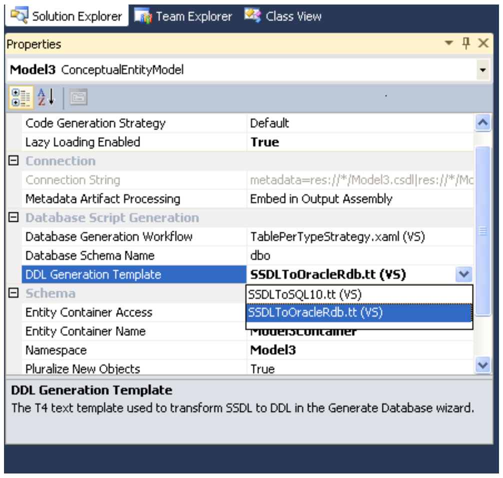
10.4 Generating Database Schema from Model

Visual Studio 2010 with Entity Framework V3.5 enables you to generate a database schema from an existing Data Entity Model.

This section will take you through the steps required to generate a fresh database schema definition from a model.

How to generate a schema from an Entity Data Model:

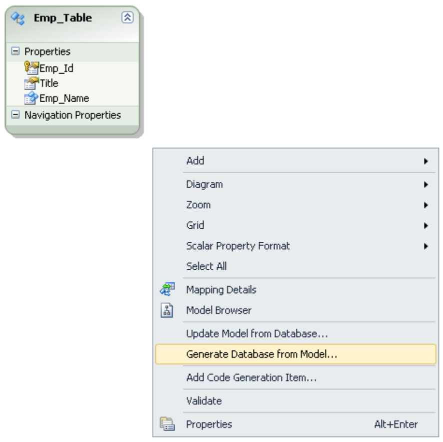
Step1: From the **Properties** window of your selected data model choose **SSDLToOracleRdb.tt** in the **DDLGeneration Template** property as shown below:



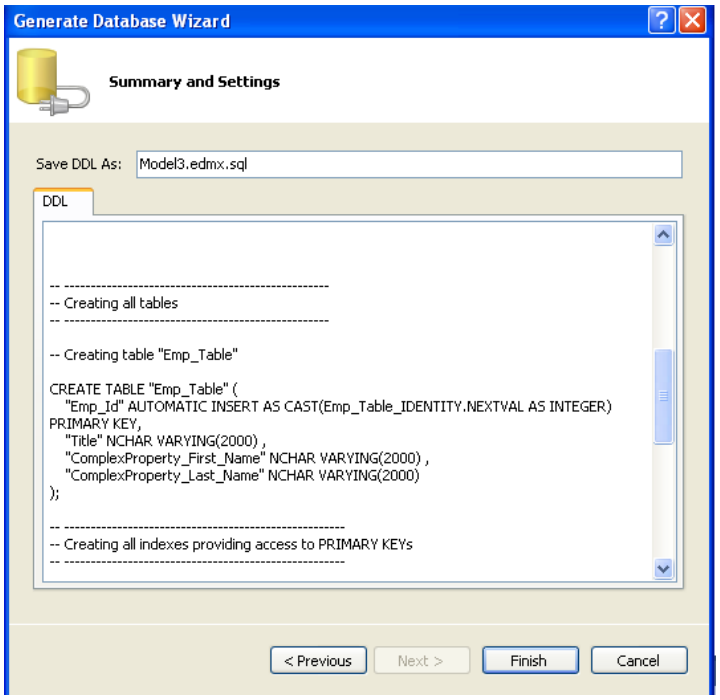
Code Generation Strategy	Default
Lazy Loading Enabled	**True**
Connection	
Connection String	metadata=res://*/Model3.csdl|res://*/Mc
Metadata Artifact Processing	Embed in Output Assembly
Database Script Generation	
Database Generation Workflow	TablePerTypeStrategy.xaml (VS)
Database Schema Name	dbo
DDL Generation Template	**SSDLToOracleRdb.tt (VS)**
Schema	SSDLToSQL10.tt (VS) / SSDLToOracleRdb.tt (VS)
Entity Container Access	
Entity Container Name	**Model3Container**
Namespace	**Model3**
Pluralize New Objects	True

DDL Generation Template
The T4 text template used to transform SSDL to DDL in the Generate Database wizard.

Step2: **Right click** on entity designer and select **Generate Database from model**



Step3: Add a Connection to the database when prompted.
Step4: Click **Next**. A new SQL script will be generated .



Save DDL As: Model3.edmx.sql

```
-- --------------------------------------------------
-- Creating all tables
-- --------------------------------------------------

-- Creating table "Emp_Table"

CREATE TABLE "Emp_Table" (
   "Emp_Id" AUTOMATIC INSERT AS CAST(Emp_Table_IDENTITY.NEXTVAL AS INTEGER)
PRIMARY KEY,
   "Title" NCHAR VARYING(2000) ,
   "ComplexProperty_First_Name" NCHAR VARYING(2000) ,
   "ComplexProperty_Last_Name" NCHAR VARYING(2000)
);

-- --------------------------------------------------
-- Creating all indexes providing access to PRIMARY KEYs
-- --------------------------------------------------
```

You can now copy the SQL script create and execute on your Rdb database using Interactive SQL.

Show Navigation

Chapter 11 Unsupported features

- Foreign Key support
- Open Table definition
- Alter Procedure
- Data View Designer
- Import/export table data as XML.

Show Navigation

Glossary

assembly
Assembly is Microsoft's term for the module that is created when a DLL or .EXE is complied by a .NET compiler.

Binary Large Object (BLOB)
A large object datatype who's content consists of binary data. Additionally, this data is considered raw as its structure is not recognized by the database.

Character Large Object (CLOB)
The LOB datatype whose value is composed of character data corresponding to the database character set.

data provider
As the term is used with Rdb Data Provider for .NET, a data provider is the connected component in the ADO.NET model and transfers data between a data source and the `DataSet`.

dirty writes
Dirty writes means writing uncommitted or dirty data.

DDL
DDL refers to data definition language, which includes statements defining or changing data structure.

DOM
Document Object Model (DOM) is an application program interface (API) for HTML and XML documents. It defines the logical structure of documents and the way that a document is accessed and manipulated.

flush
Flush or flushing refers to recording changes (that is, sending modified data) to the database.

instantiate
A term used in object-based languages such as C# to refer to the creation of an object of a specific class.

Large Object (LOB)
The class of SQL datatype that is further divided into internal LOBs and external LOBs. Internal LOBs include `BLOB`s, `CLOB`s, and `NCLOB`s while external LOBs include `BFILE`s.

Microsoft .NET Framework Class Library
The Microsoft .NET Framework Class Library provides the classes for the .NET framework model.

namespace

- .NET:

  A namespace is naming device for grouping related types. More than one namespace can be contained in an assembly.

- XML Documents:

  A namespace describes a set of related element names or attributes within an XML document.

National Character Large Object (NCLOB)
The LOB datatype whose value is composed of character data corresponding to the database national character set.

octet
An 8-bit unit, usually referred to as BYTE

RdbDataReader
An `RdbDataReader` is a read-only, forward-only result set.

primary key
The column or set of columns included in the definition of a table's PRIMARY KEY constraint.

reference semantics
Reference semantics indicates that assignment is to a reference (an address such as a pointer) rather than to a value. See **value semantics**.

result set
The output of a SQL query, consisting of one or more rows of data.

savepoint
A point in the workspace to which operations can be rolled back.

stored procedure
A stored procedure is a block of SQL code that Rdb stores in the database and can be executed from an application.

Unicode
Unicode is a universal encoded character set that enables information from any language to be stored using a single character set.

URL
URL (Universal Resource Locator).

value semantics
Value semantics indicates that assignment copies the value, not the reference or address (such as a pointer). See **reference semantics**.

Show Navigation