

DEVELOPER: .Net

It Takes All Types
By Christian Shay



Build applications with Oracle Developer Tools for Visual Studio and Oracle user-defined types.

Using Oracle Data Provider for .NET (ODP.NET), a high-performance ADO.NET-compliant data provider, developers can take advantage of all of Oracle Database's enterprise features with performance tuning and security features built in. ODP.NET can be used in conjunction with Oracle Developer Tools for Visual Studio, an add-in for Microsoft Visual Studio that makes common database tasks easy for .NET developers. For example, you can create tables, edit and debug PL/SQL, edit and run SQL scripts, automatically generate .NET code to access database schema objects, and do many other database tasks right from Visual Studio.

User-defined types (UDTs) make it possible to model real-world entities such as customers or purchase orders as objects in Oracle Database, and ODP.NET 11g has added support for UDTs, so that developers can access their objects, VARRAYs, and nested table types from C# or VB.NET code. Oracle Developer Tools for Visual Studio also supports UDTs, enabling you to use Visual Studio to create and modify them and autogenerate .NET code to access them. (UDTs will be of particular interest to .NET developers leveraging Oracle Spatial and Oracle Advanced Queuing, which accept UDTs as part of their PL/SQL interfaces. VARRAYs and nested table types are frequently used in PL/SQL business logic.)

In this article, I use both ODP.NET and Oracle Developer Tools for Visual Studio to take you, step by step, through the process of using Visual Studio to create UDTs in Oracle Database, and then I create a sample ASP.NET Web application to display the value of those objects.

The steps in this article require Visual Studio 2008 or Visual Studio 2005 and access to an Oracle Database (Oracle9i Database or later) instance. You should also be able to get this article's sample application working with Visual Studio .NET 2003, in which case the steps will be significantly different from those used in this article.

You'll need to download and install both Oracle Developer Tools for Visual Studio and ODP.NET version 11.1 or later. These products are free and can be downloaded from the Oracle Technology Network using the link in "Next Steps," at the end of this article. If you are not familiar with the object-relational features of Oracle Database, see "Next Steps" to find the URL for accessing Oracle Database Object-Relational Developer's Guide, which describes concepts and terminology used in this article.

Designing the Data and the Application

I keep the database design simple while demonstrating some of the object-relational features of Oracle Database. To that end, I first define a simple customer object type, CUSTOMER_OBJ, that consists of a customer number; a customer name; an address object, ADDRESS_OBJ; and a phone number list, PHONELIST_OBJ (which will be a VARRAY of as many as 10 phone numbers). The CUSTOMER_OBJ objects will live in the CUSTOMER_TAB object table. (When you store objects in an object table, they become referenceable, meaning that you can refer to them via a pointer, a REF, in other table columns.

As Published In

ORACLE
MAGAZINE
May/June 2008

TAGS

dotnet, All

£75K+ Technology Jobs

Thousands of UK Tech Jobs. Search & Apply Online Today!

www.TechnologyLadder.co.uk

The Oracle SOA Experts

SOA Consultants, SOA Training, SOA Strategies, SOA for Apps

www.griffiths-waite.co.uk

IT Jobs in Banking

IT recruitment in banking & IT jobs in the financial markets

www.eFinancialCareers.co.uk

OracleFinancials

StayAhead Training offers training in the OracleFinancials suite

www.stayahead.com

UK Oracle Consultants

The leading Oracle eBusiness suite consultants in the UK

www.pdg-consulting.com

You can also "dereference" any REF in SQL statements and in application code.) Finally, I create a CUSTOMER_MEETINGS relational table that consists of a meeting ID, a meeting date, a location, and a REF to a CUSTOMER_OBJ object.

My ASP.NET Web application design is also very simple: I display all customer meetings on a Web page, along with all of the details for each customer. I could have just as easily created a Winform (thick client) application, and the steps to do so are very similar to what I do here for this ASP.NET Web application.

Creating User-Defined Types

Oracle Database's primary integration point with Visual Studio is server explorer (see Figure 1), a tree control that enables you to view your Oracle Database schema. Each schema object in server explorer provides context menu items that launch various wizards and designers.

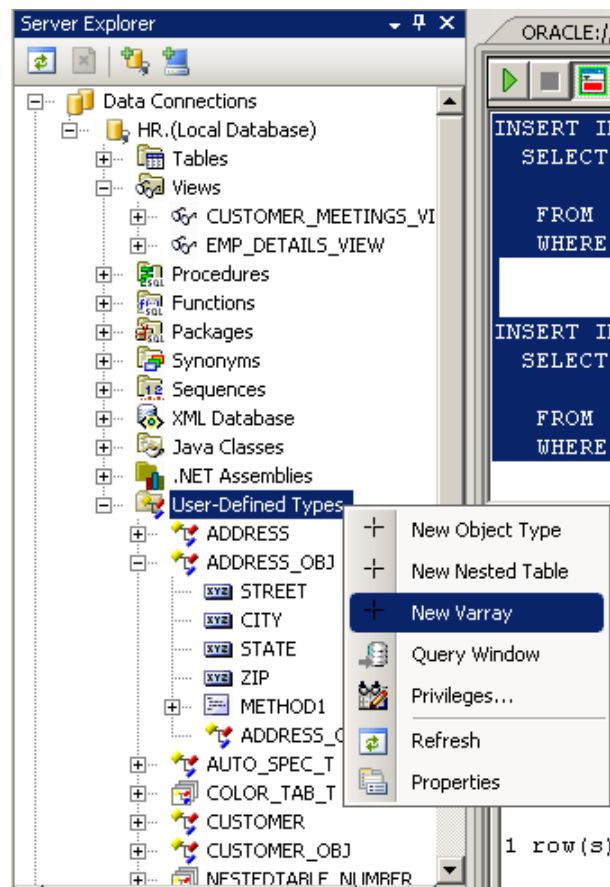


Figure 1: Server explorer and context menu

With Visual Studio, ODP.NET, and Oracle Developer Tools for Visual Studio installed and an Oracle Database schema available, you are ready to start building the application. First, connect to Oracle Database. In Visual Studio, right-click the Data Connections node of server explorer and select Add Connection. Make sure that Oracle Data Provider for .NET is selected, fill in the connection dialog box with the necessary database information, and click OK to connect.

Next, construct the various child components of the CUSTOMER_OBJ object, starting with the address object type. In server explorer, scroll down to the User-Defined Types node, right-click it, and select New Object Type. This launches object designer, as shown in Figure 2. Enter ADDRESS_OBJ for Type name. Click Add four times to create four attributes. Select each attribute on the left side of the designer (under Attributes) and then change its Name and Type on the right side (under Attribute Properties) using the following values:

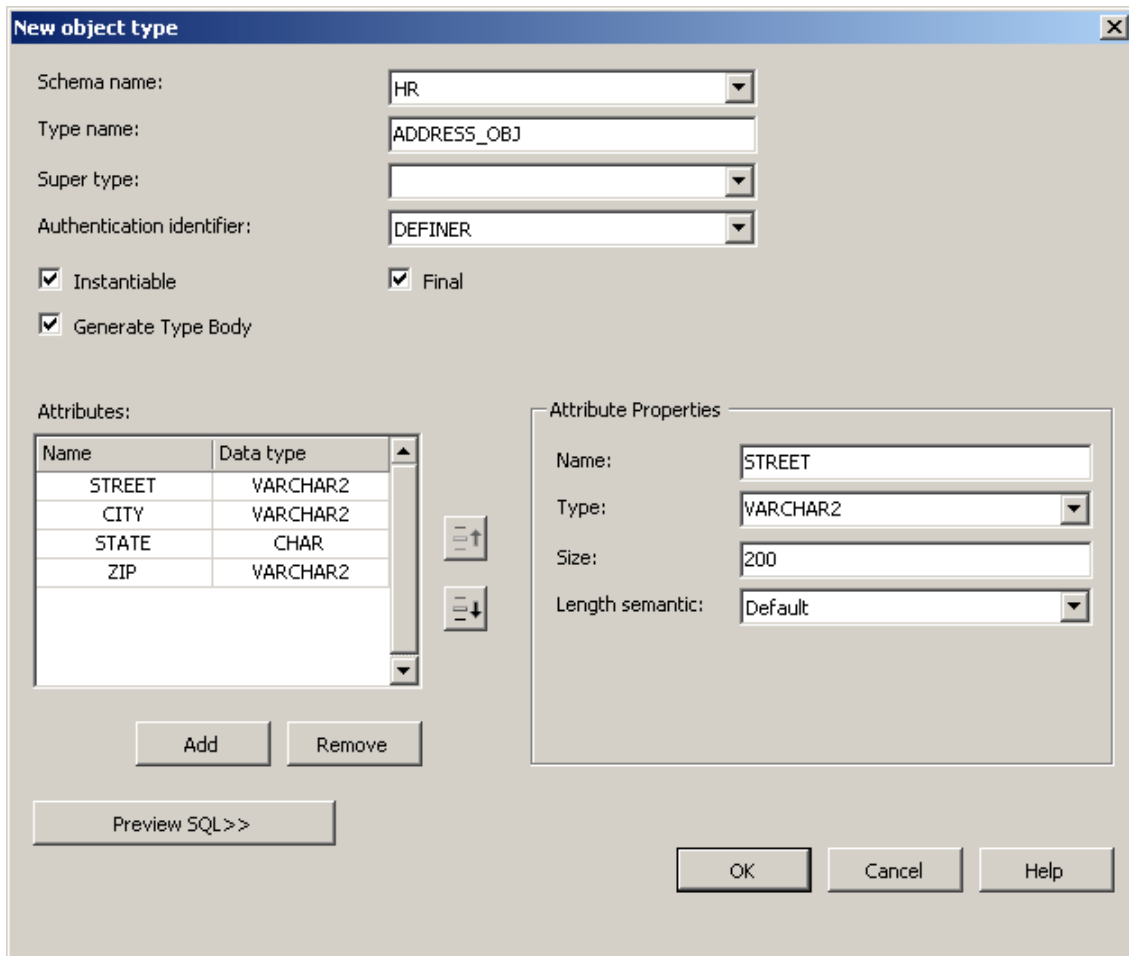


Figure 2: Object designer

Name	Type (Data type)
STREET	VARCHAR2 (200)
CITY	VARCHAR2 (200)
STATE	CHAR(2)
ZIP	VARCHAR2 (20)

Note that you can click Preview SQL>> to see the SQL that is going to be executed. Click OK to create the type.

Now, create PHONELIST_OBJ, which is a VARRAY of phone numbers. Again right-click the User-Defined Types node, and this time select New Varray (as shown in Figure 1). This launches the VARRAY designer. In the Type Name field, enter PHONELIST_OBJ; in the Limit field, enter 10; in the Type field, enter `varchar2`; and in the Size field, enter 20. Then click OK to create the object type.

The last type to define is CUSTOMER_OBJ. Once again, right click the User-Defined Types node and select New Object Type.

Enter CUSTOMER_OBJ for Type name. As you did when creating ADDRESS_OBJ, first click Add four times to create four attributes. Next, select each attribute on the left side of the designer (under Attributes) and then change its Name and Type on the right side (under Attribute Properties), using the following

values:

Name	Type (Data type)
CUSTNO	NUMBER
CUSTNAME	VARCHAR2(200)
ADDRESS	ADDRESS_OBJ
PHONELIST	PHONELIST_OBJ

The last two attributes are user-defined types you just created. You will find user-defined types at the bottom of the Attribute Properties Type list, after all of the standard Oracle datatypes.

Now you need to create an object table to give the CUSTOMER_OBJ objects a place to live, so that you can refer to them in other tables by using REFS. In server explorer, right-click the Tables node and select New Object Table. When the dialog box opens, enter CUSTOMER_TAB for Table name and select CUSTOMER_OBJ from the Object type list (see Figure 3). Click Save to create the table.

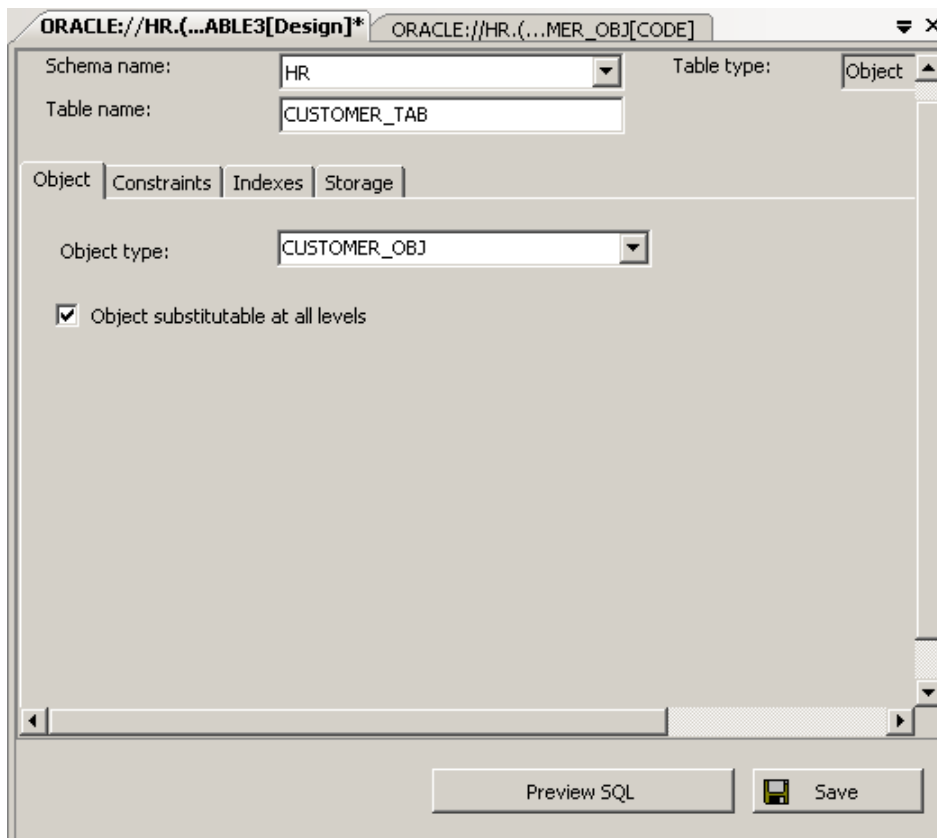


Figure 3: Object table designer

To finish building your database schema objects, you need to create the CUSTOMER_MEETINGS table. In server explorer, right-click the Tables node and select New Relational Table. Enter CUSTOMER_MEETINGS for Table name. Click Add four times to create four columns. Next, select each column on the left side of the designer (under Columns) and then change its Name and Data type on the right side (under Column Properties) using the following values:

Name	Data type
MEETING_ID	NUMBER
TIME	DATE


```

FROM Customer_tab C
WHERE C.CustNo = 1 ;

INSERT INTO Customer_meetings
SELECT '2', '13-JAN-2008', 'HQ - 300', REF(C)
FROM Customer_tab C
WHERE C.CustNo = 2 ;

```

Creating the ASP.NET Web Application

Now that the database is ready, create the ASP.NET Web application, by taking advantage of the automatic .NET code generation features of Visual Studio. In the main menu, choose File -> New Project, and then under Visual C#, select ASP.NET Web Application. Choose View -> Toolbox, and then under the Data section of the Toolbox, drag and drop a GridView control onto the ASP.NET design surface. Right-click the small arrow in the upper right-hand corner of the GridView control to open the GridView Tasks pane (see Figure 5). From the Choose Data Source list, select <New Data Source>. This will start the datasource configuration wizard, which will create and configure a SQLDataSource component that contains automatically generated ODP.NET data access code. When the wizard opens, double-click the database icon. Select the correct connection, and then continue to the "Configure the Select Statement" screen. Click the Specify a Custom SQL Statement radio button, click Next, and enter the following SQL statement in the text box:

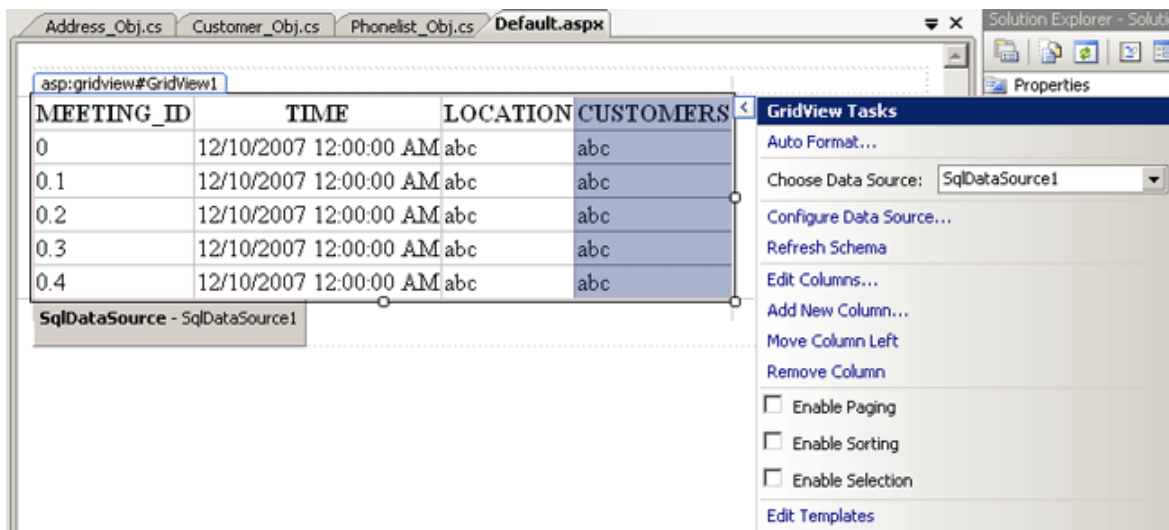


Figure 5: Configuring the GridView control

```

SELECT MEETING_ID, TIME, LOCATION, Deref(CUSTOMER)
FROM CUSTOMER_MEETINGS

```

Note that the SQL includes the SQL Deref operator. This makes sure that the object value is returned, rather than the REF, which is nothing more than a large hexadecimal value, similar to a ROWID. (You could also create an application that queries for the REF and then fetches each object value individually, using its REF, but that approach is not covered in this article.)

Click Next and then Test Query to make sure there are no errors. Then click Finish. Go to the GridView Tasks pane again, and click Add New Column. For Header Text, enter *Customers*, and for Data Field, select Deref(CUSTOMER).

Using the Custom Class Wizard

At this point, the GridView and SQLDataSource controls do not know what to make of the CUSTOMER_OBJ types being returned by Oracle Database. By definition, they are user-defined, and

Next Steps

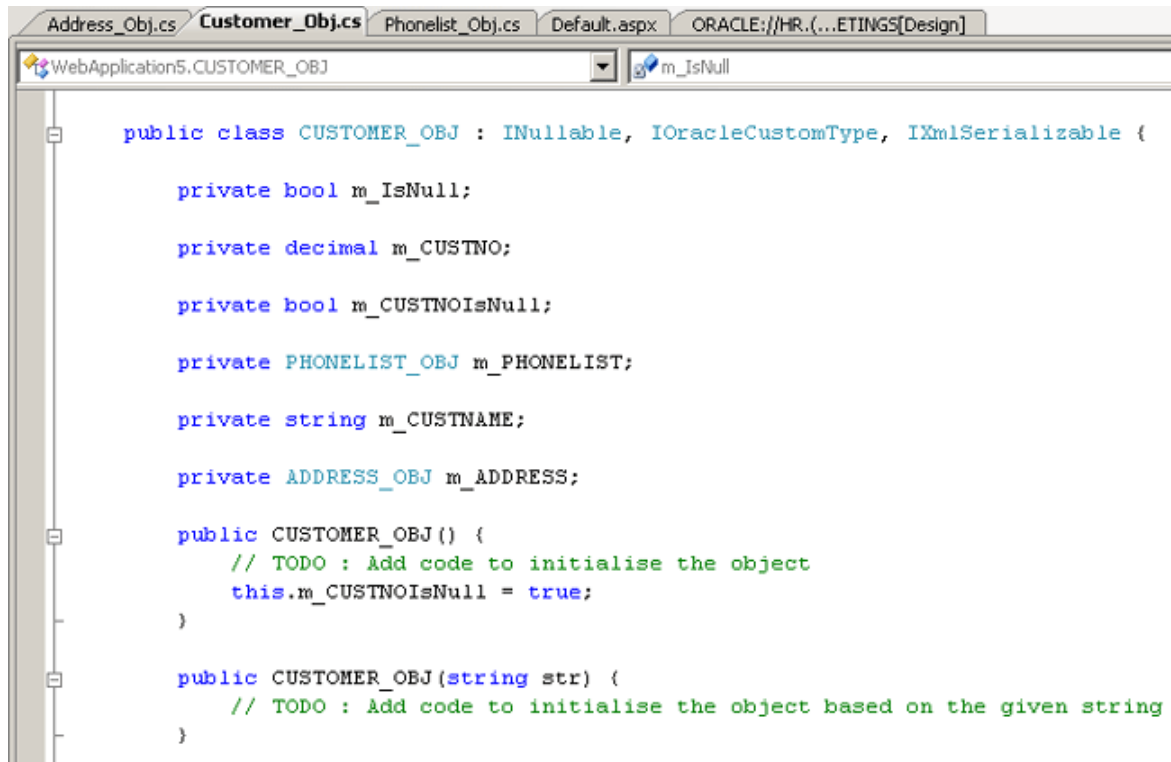
[DOWNLOAD ODP.NET and Oracle Developer Tools for Visual Studio .NET 11g](#)

READ more [Shay](#)

READ more about object-relational features [Oracle Database Object-Relational Developer's Guide](#)

therefore you must provide a mapping between the Oracle user-defined datatype and a .NET class that will hold the data. To do this, use the Oracle Custom Class wizard.

In server explorer, go to each of the three user-defined types you created (CUSTOMER_OBJ, PHONELIST_OBJ, and ADDRESS_OBJ) and do the following: Right-click the type name, and select Generate Custom Class. Then accept all the defaults until you reach the end of the wizard, which will add a C# class file to the project. Figure 6 shows how a generated custom class (Customer_obj.cs) maps to the CUSTOMER_OBJ Oracle Database UDT.



```

public class CUSTOMER_OBJ : INullable, IOracleCustomType, IXmlSerializable {

    private bool m_IsNull;

    private decimal m_CUSTNO;

    private bool m_CUSTNOIsNull;

    private PHONELIST_OBJ m_PHONELIST;

    private string m_CUSTNAME;

    private ADDRESS_OBJ m_ADDRESS;

    public CUSTOMER_OBJ() {
        // TODO : Add code to initialise the object
        this.m_CUSTNOIsNull = true;
    }

    public CUSTOMER_OBJ(string str) {
        // TODO : Add code to initialise the object based on the given string
    }
}

```

Figure 6: Custom-generated class for CUSTOMER_OBJ

The SQLDataSource control will ask the CUSTOMER_OBJ object to render its value as a string, by calling its ToString() method, so you need to modify the ToString() method on both the CUSTOMER_OBJ and the PHONELIST_OBJ custom classes. (In this example, you can ignore the ADDRESS_OBJ ToString() method, because its individual members are rendered explicitly by the CUSTOMER_OBJ ToString() method). For this sample application, use the code in Listing 2 for the ToString() methods.

Code Listing 2: ToString() method for custom classes

ToString() method for CUSTOMER_OBJ class (in Customer_obj.cs):

```

public override string ToString() {
return "Customer Number: " + CUSTNO + "<br>Name: " + CUSTNAME + " <br>Address: " +
ADDRESS.STREET + " " + ADDRESS.CITY + " " + ADDRESS.STATE+ " " + ADDRESS.ZIP +
"<br>Phone Numbers:<br>" + PHONELIST.ToString(); }

```

ToString() method for PHONELIST_OBJ class (in Phonenumber_obj.cs):

```

public override string ToString()
{
    string ret = "";
    for (int x = 0; x < m_PHONELIST_OBJ.Length; x++)
    {
        ret = ret + m_PHONELIST_OBJ[x] + "<br>";
    }
    return ret;
}

```

One last thing you need to do is make sure the HTML tags you are passing back with the

CUSTOMER_OBJ custom class ToString() method are honored by the GridView control. Go back to GridView Tasks, and select Edit Columns. In the dialog box that appears, under Selected Fields, select CUSTOMERS. In the Properties pane on the right, set HTML Encode to False and then click OK.

Running the ASP.NET Web Application

From the main menu, select Debug -> Start without Debugging. This starts a Microsoft Internet Information Server Web server (built in to Visual Studio) and also launches a Web browser with a URL pointing to that Web server. The result is shown in Figure 7. The Web browser displays a grid containing the MEETING_ID, TIME, LOCATION, and CUSTOMERS column data.

MEETING_ID	TIME	LOCATION	CUSTOMERS
1	1/12/2008 12:00:00 AM	HQ - 200	Customer Number: 1 Name: John Doe Address: 111 Lollypop Lane Redwood Shores CA 95054 Phone Numbers: 415-555-1212
2	1/13/2008 12:00:00 AM	HQ - 300	Customer Number: 2 Name: John Smith Address: 333 Island Drive Edison NJ 08820 Phone Numbers: 609-555-1212 201-555-1212
3	1/13/2008 12:00:00 AM	HQ - 200	Customer Number: 3 Name: Richard Roe Address: 555 Drive Edison NJ 08820 Phone Numbers: 415-555-1212 650-555-1212 510-555-1212

Figure 7: Running the application

Summary

Using Visual Studio, you have created user-defined types in Oracle Database, an object table, and a relational table with a REF to our Customer object. You populated the objects with data, generated .NET custom classes for our objects, customized the string output of those classes, and then rapidly created a Web application to display the data in a Web browser.

Now that you have walked through the basics, I encourage you to further exercise the ODP.NET and Oracle Developer Tools for Visual Studio 11g releases. Happy coding!

Christian Shay (christian.shay@oracle.com) is a principal product manager at Oracle.

[Send us your comments](#)