GETTING STARTED WITH ASP.NET TO ORACLE 11G USING THE ADO.NET ENTITY DATA FRAMEWORK
INTRODUCTION

In this tutorial, we demonstrate how you can easily connect Microsoft Visual Studio to an Oracle 11g server by using the Progress DataDirect Connect for ADO.NET in conjunction with the ADO.NET Entity Framework. This configuration lets you fully exploit the power of Microsoft’s ADO.NET Entity Data Framework when working with Oracle without needing to write any code.

> Visual Studio setup
> Install and test DataDirect Connect for ADO.NET
> Create a new Web application project
> Build the user interface
> Get a list of job titles
> Retrieve a list of all employees
> Display details for a selected employee

While new architectures that interact with data are continually arising, one constant remains: the underlying technologies that make access to information possible remain as critical as ever. In this article, you’ll see that all communication with the database is handled within the CLR (Common Language Runtime), leading to better performance, security, and reliability. What makes DataDirect Connect for ADO.NET such a compelling choice for this task is that its 100% managed code architecture eliminates the need to install, configure, or maintain any additional client-side software such as Oracle’s client libraries, as shown in figure 1.

![Figure 1](image-url)
You’ve probably heard a lot about the ADO.NET Entity Framework. The purpose of this article is to help you make use of it in your Oracle environment.

The example for this article is a simple Web-based application that presents a basic employee directory using sample data from an Oracle 11g instance. While this simple application and its rudimentary user interface won’t win any GUI design awards, you will learn how to get productive quickly and without needing to write any code. **Click here to download the sample project.**

**INSTRUCTIONS**

1. **Install Visual Studio Professional 2008 or Team System Edition 2008.**
   You’ll need Service Pack 1 for either product.

   If you already have Visual Studio on your computer, make sure to download the .NET Framework 3.5 Service Pack 1: the wizards and other key components won’t work without it.

2. **Install DataDirect Connect for ADO.NET**

   Please note that this article was written with version 3.3. If you are using a newer version, you can follow these same steps as well

   Make sure that you have administrative privileges on this computer. In addition, before installing the DataDirect product, make sure you’re running one of the following operating systems:

   - Windows Server 2008 (all editions)
   - Windows Vista (all editions)
   - Windows XP Home Edition and Professional
   - Windows Server 2003 (all editions)
   - Windows 2000
DataDirect Connect for ADO.NET supports the 32 and 64 bit versions of these operating systems. Don’t worry: the installer takes care of this for you automatically.


Make sure that you’ve selected .NET Framework 3.5 from the drop-down on the upper right side of the New Project dialog box.

4. Add the ADO.NET Data Model to your project by following these steps:

   a. Right-click on the name of your new project.

   b. Choose Add-> New Item.

   c. Select the ADO.NET Entity Data Model template. This launches the Entity Data Model Wizard.

   d. Choose the Generate from database option.

   e. Click on the New Connection button.

      a. Fill in details about your Oracle connection as shown in figure 2. To access the HR schema, use the scott/tiger login. In addition, make sure to select the DataDirect Connect for ADO.NET Oracle Data Provider.
When you’ve finished, click on the Test Connection button to double-check that everything is configured correctly. If you receive a Test connection succeeded message, move on to these steps:

a. Click the Yes, include the sensitive data in the connection string radio button.

b. Check the Save entity connection settings in App.Config box, and enter Entities in the text box as shown in figure 3.

c. Click Next. The Wizard connects to the database and presents a list of the objects that you can include in your model.

d. Expand the Tables entry, and place check marks next to the EMPLOYEES (HR) and JOBS (HR) entries.

e. Click Finish.
The Wizard now connects to Oracle, deciphers the relationships between these two tables, and then creates a data model based on what it learned. There are a few important points to bear in mind about the interaction between the Entity Data Model Wizard and Oracle 11g:

- It may take a few minutes for the Wizard to complete its work.
- The Wizard must be able to infer primary and foreign keys from your Oracle schema.
- Be on the lookout for any diagnostic messages from Entity Framework and related Wizard regarding your database structure.
- If your Oracle tables use niche data types, such as UDT (user-defined types), you may receive some messages from the Entity Framework.
- The Entity Framework mandates that your stored procedure parameters need to be comprised of supported data types.

When the Wizard completes its work, click the Show all Files icon at the top of the Solution Explorer. Notice that the Wizard has created numerous references, an App.Config file, and a data model.

Figure 4 shows the relationship mapping between the two tables as visualized in the ADO.NET Entity Data Model Designer, the references and other files that were created, as well the Model Browser:
5. Build and save your project.

To build the project, right click on the project name in the Solution Explorer and choose Build. In the upcoming steps, you’ll construct the ASP.NET application without writing any code.

It’s now time to build the user interface, which consists of adding a series of GUI controls and related data sources.

6. Add a DropDownList control to the application.

The first step in building the user interface is to create a control on the page that selects a department. Here’s how to do that:

a. Switch to the Default.aspx page tab.

b. Switch to the Design view at the bottom of the screen.

c. Make the toolbox visible (as shown in figure 5), and drag a DropDownList control onto the blank page. Once configured, this will provide a list of job titles from Oracle; we’ll then retrieve a list of employees for the chosen department.

d. Check the Enable AutoPostBack box on this control.

e. In the Properties window, rename this control to JobTitles. You’ll find this setting in the field.

f. Build and save the project, but don’t run it yet. You must first associate a data source with the control, which comes next.

7. Get a list of job titles from the Oracle database.

To do this, we need to connect a data source with the control you just added. Here’s how:

a. Drag an EntityDataSource control from the toolbox onto the page.

b. Click the flyout on the top right of the control and choose Configure Data Source.

c. Choose the Named Connection and DefaultContainerName you created earlier in step 4, and click Next.
d. Choose the JOBS value from the EntitySetName drop-down box.

e. Leave the EntityTypeFilter set to the default of (None).

f. Check the Select All (Entity Value) box and click Finish. Open the Properties window for the EntityDataSource and set the ID property to DataSource.JobTitle, which renames the data source to something more meaningful.

g. Click on the drop-down list you created in step 6. Select the Choose Data Source menu option.

h. When the Data Source Configuration Wizard pops up, click on the Refresh Schema link at the bottom of the window.

i. Select the DataSourceJobTitle from the Select a data source drop-down.

j. Choose JOB_TITLE in the Select a data field to display in the DropDownList field. This shows the user the actual name of the job.

k. Choose JOB_ID in the Select a data field for the value of DropDownList field and click OK as shown in figure 6. This returns the internal identifier for the selected job.

l. Once you’ve finished, save your project and then click the green Start Debugging icon at the top of the main Visual Studio window. Make sure to accept the Modify the Web.config file to enable debugging radio button. You should now have a working Web page with a populated drop-down list of job titles. It’s time to tie this together with employee details.
8. Retrieve a list of all employees with the job title you choose from the drop-down list. In this step, we take advantage of the Entity Framework’s understanding of the inherent relationships among our tables. Here are the steps to follow:

   a. To space things out on the page, hit Return 2 times beneath the DataSource.JobTitle control.

   b. Drag a GridView control onto the page, followed by an EntityDataSource. We’ll use the first control to display the employee list, and the second one to connect the control to the database.

   c. Rename the EntityDataSource to DataSourceEmployeeList by changing the value of the ID field on its Properties window.

   d. Click the flyout on this data source, and choose the Configure Data Source menu option.

   e. Choose the Named Connection and DefaultContainerName you created earlier in step 4, and click Next.

![Figure 6](image-url)
f. Choose EMPLOYEES from the EntitySetName, and check the Select All (Entity Value) box.

g. Click Finish to complete selecting data.

Next up, it’s time to set the WHERE clause for the query. Here’s where we’ll connect the job title list from earlier with related employees; remember that this relationship was discovered when we used the Wizard to explore the database. First up is retrieving data from Oracle:

a. Open the Properties window for DataSourceEmployeeList, and click on the ellipses next to the Where entry.

b. Check the Automatically generate the Where expression based on the provided parameters box.

c. Click on the Add Parameter button.

d. Rename the new parameter entry JOBS.JOB_ID. This takes advantage of the relationship that the entity framework discovered between the EMPLOYEES and JOBS tables.

e. Set the Parameter source drop-down to Control.

f. Set the ControlID drop-down to JobTitles. This is the control that returns the job title and job id. Figure 9 shows how this dialog should appear:

![Figure 7](image-url)
g. Click OK to save your changes to the data source.

After this, we need to associate the grid of employees with the Oracle data:

a. Click on the grid you dragged onto the page two sequences ago. Open its Properties window, and rename it to EmployeesGrid.

b. Bring up the GridView Tasks dialog by clicking on the grid, and select the DataSourceEmployeeList for the Choose Data Source drop-down.

c. Check the Enable Selection box. This will let us choose a particular employee for further analysis.

d. Save project, and then run it in debug mode. You should see changes in the drop-down reflect in the employee list, just like in figure 8:

![Figure 8](image-url)
9. Display details for a selected employee.

   Now that we’ve retrieved a list of all employees with this job title, it’s
time to drill-down and see all of the details of a selected employee.

   a. To space things out on the page, hit Return 2 times beneath the
EmployeesGrid control.

   b. Drag a DetailsView control onto the page. Notice that instead of
displaying a list of items, this control only displays one.

   c. Drag one more EntityDataSource onto the page.

   d. Rename the just-added EntityDataSource to
DataSourceEmployeeDetail.

   e. Open the Properties window for DataSourceEmployeeDetail, and
click on the ellipses next to the Where entry.

   f. Check the Automatically generate the Where expression based on
the provided parameters box.

   g. Click on the Add Parameter button.

   h. Rename the new parameter entry EMPLOYEE_ID.

   i. Set the Parameter source drop-down to Control.

   j. Set the ControlID drop-down to EmployeesGrid. In other words,
clicking on a row in the grid will populate this data source.

   k. Click OK to save.

The last step is to associate the user interface with its data source:

   a. Click on the DetailsView control you added in the previous
sequence.

   b. Change to its Properties window, and rename it to
EmployeeDetail.

   c. Click on the flyout next to the control, and select the
DataSourceEmployeeDetail data source.
Save the project, and then run it in debug mode. Your results should appear similar to figure 9.

Now that you have a functioning example of all these technologies working in concert, it's easy to simply switch out your DataDirect Oracle provider with a DataDirect provider for another database brand (such as SQL Server, Sybase, or DB2). The code will work equally well with those products (provided, of course, that you create similar tables in those databases and the providers themselves support the Entity Framework).

AUTHOR

Robert Schneider is a Silicon Valley-based technology consultant. He has written five books and numerous articles on advanced technical topics such as Service Oriented Architecture (SOA), open source, and relational database design/optimization. He can be reached at Robert.Schneider@think88.com.
PROGRESS SOFTWARE

Progress Software Corporation (NASDAQ: PRGS) is a global software company that enables enterprises to be operationally responsive to changing conditions and customer interactions as they occur. Our goal is to enable our customers to capitalize on new opportunities, drive greater efficiencies, and reduce risk. Progress offers a comprehensive portfolio of best-in-class infrastructure software spanning event-driven visibility and real-time response, open integration, data access and integration, and application development and management—all supporting on-premises and SaaS/cloud deployments. Progress maximizes the benefits of operational responsiveness while minimizing IT complexity and total cost of ownership.

WORLDWIDE HEADQUARTERS

Progress Software Corporation, 14 Oak Park, Bedford, MA 01730 USA
Tel: +1 781 280-4000  Fax: +1 781 280-4095  On the Web at: www.progress.com

For regional international office locations and contact information, please refer to the Web page below:
www.progress.com/worldwide

Progress, DataDirect, and Business Making Progress are trademarks or registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and other countries. Any other trademarks contained herein are the property of their respective owners. Specifications subject to change without notice.

© 2010 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.
Rev. 07/10  8525-131845