**ASP.NET 4.0 与 Entity Framework 4-第三篇-使用Entity Framework调用存储过程**

LloydSheng 写于 2010年04月26日下午 3:34 | [有5人评论](http://lloydsheng.com/2010/04/aspnet40-entityframework4-execute-stored-procedures-using-entity-framework4.html#comments)

**摘要**

本文将向你介绍如何调用存储过程对SQL Server数据库中的数据进行CRUD操作。文中采用的数据库依然是我们在本系列第一篇文章《[采用Model-First 开发方式创建数据库](http://lloydsheng.com/2010/04/aspnet-40-and-entity-framework4-part1-create-database-using-modelfirst-development.html)》中创建的，使用的Web页面是在第二篇文章《[使用Entity Framework 进行CRUD操作](http://lloydsheng.com/2010/04/aspnet40-entityframework4-how-to-crud-using-entityframework.html)》中创建的 ，可以点击[[代码](http://files.cnblogs.com/lloydsheng/OrderSystem2.zip)]下载链下载项目文件，要用VS2010打开。

**步骤1.创建存储过程**

首先需要创建存储过程，第一个存储过程的功能是查表UserAccount记录的，代码如下：

CREATE PROCEDURE dbo.UserAccounts\_SelectAll

AS

SET NOCOUNT ON

SELECT Id, FirstName, LastName, AuditFields\_InsertDate,

AuditFields\_UpdateDate

FROM UserAccounts

RETURN

下面一个存储过程功能是通过Id选择UserAccount表中一条记录，代码如下：

CREATE PROCEDURE dbo.UserAccounts\_SelectById

(

@Id int

)

AS

SET NOCOUNT ON

SELECT Id, FirstName, LastName, AuditFields\_InsertDate,

AuditFields\_UpdateDate

FROM UserAccounts

WHERE Id = @Id

RETURN

第三个存储过程的功能是向UserAccounts表中插入一条记录，代码如下：

CREATE PROCEDURE dbo.UserAccounts\_Insert

(

@FirstName nvarchar(50),

@LastName nvarchar(50),

@AuditFields\_InsertDate datetime,

@AuditFields\_UpdateDate datetime

)

AS

INSERT INTO UserAccounts (FirstName, LastName, AuditFields\_InsertDate,

AuditFields\_UpdateDate)

VALUES (@FirstName, @LastName, @AuditFields\_InsertDate,

@AuditFields\_UpdateDate)

SELECT SCOPE\_IDENTITY() AS Id

第四个存储过程的功能是更新UserAccount表中的数据，代码如下：

CREATE PROCEDURE dbo.UserAccounts\_Update

(

@Id int,

@FirstName nvarchar(50),

@LastName nvarchar(50),

@AuditFields\_UpdateDate datetime

)

AS

SET NOCOUNT ON

UPDATE UserAccounts

SET FirstName = @FirstName,

LastName = @LastName,

AuditFields\_UpdateDate = @AuditFields\_UpdateDate

WHERE Id = @Id

RETURN

最后一个存储过程的功能是删除UserAccount表中的记录，代码如下：

CREATE PROCEDURE dbo.UserAccounts\_Delete

(

@Id int

)

AS

SET NOCOUNT ON

DELETE

FROM UserAccounts

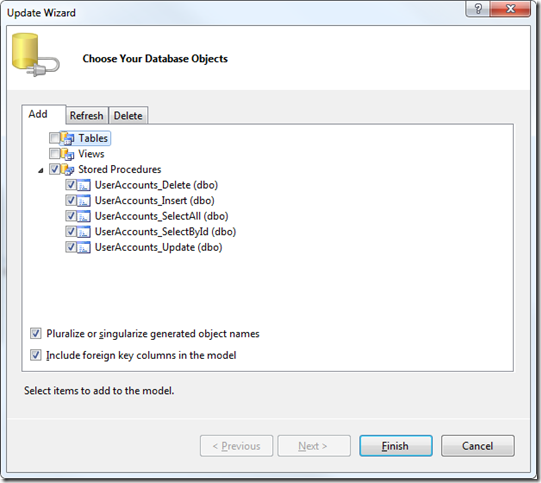
WHERE Id = @Id

RETURN

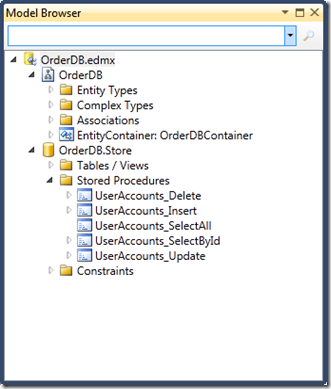
在我们进行下一步之前，你必须在OrderSystem数据库中创建上面的那些存储过程。

**步骤2:将存储过程添加到实体中**

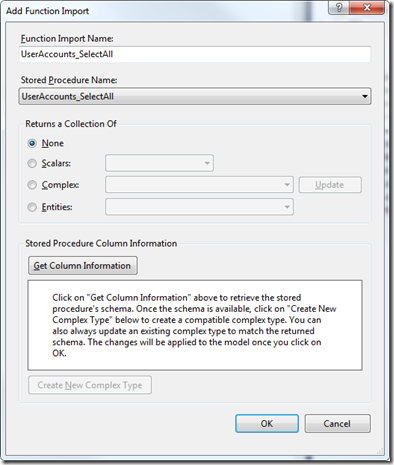
打开项目中的OrderDB.edmx文件。打开后，会出现设计器，同时模型浏览器里可以看到数据库的实体，复杂属性和关系。在OrderDB.edmx右键选择Update Mode From DataBase，这时会出现更新向导，可以看到前面的创建的5个存储过程显示出来。

[](http://lloydsheng.com/wp-content/uploads/2010/04/image.png)

展开Stored Procedures节点，选择所有存储过程，然后点击完成。这时模型浏览器中会多出5个存储过程对象。

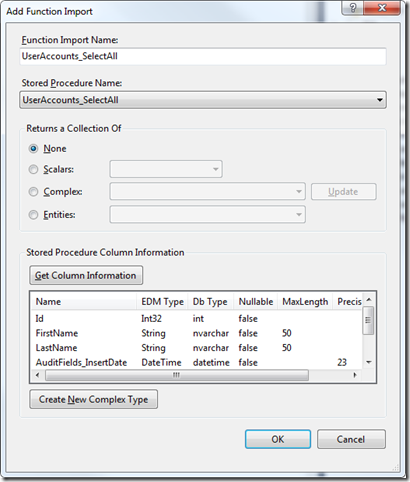
[](http://lloydsheng.com/wp-content/uploads/2010/04/image1.png)

右击模型浏览器中的UserAccounts\_SelectAll存储过程选择Add Function Import，这时就可以给OrdersDBContainer类添加一个执行UserAccounts\_SelectAll存储过程的方法了。

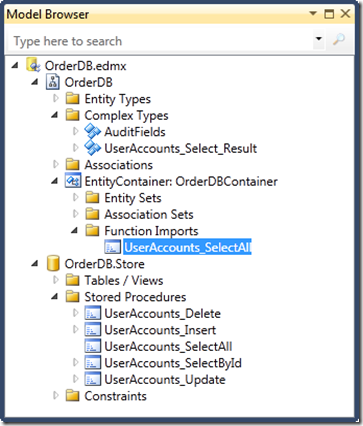
[](http://lloydsheng.com/wp-content/uploads/2010/04/image2.png)

选择存储过程执行后的返回值类型，当我们要返回一个UserAccount列表，而Framework没有提供这种类型，这时采用这个方法创建会很方便。我们可以先创建一个复杂类型用来表示存储过程返回值的类型。由于SelectAll 和 SelectById 两个存储过程返回的字段相同，只需创建一种复杂类型。

点击Get Column Information按钮，VS将会检测存储过程返回的字段。当返回字段检测完后，点击下面的Create New Complex Type 按钮，这时会自动选择Complex 选项，复杂类型的名字也会添加到右边下列框中。默认的复杂类型的名字为存储过程的名字加上“Result”。由于我将会在多个存储过程返回值中使用该复杂类型，建议取一个比较通用的名字，这里将名字改为“UserAccounts\_Select\_Result”，点击OK。

[](http://lloydsheng.com/wp-content/uploads/2010/04/image3.png)

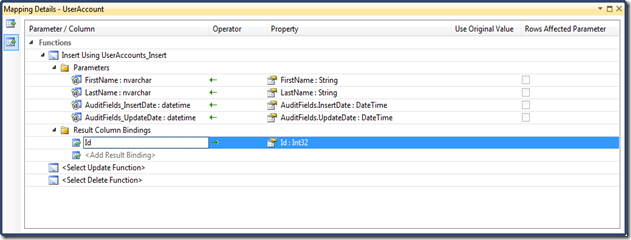
这时你会看到在模型浏览器的Function Imports 文件夹中多了一个UserAccounts\_SelectAll方法，同时在复杂类型节点下面多了一个UserAccounts\_Select\_Result类型。

[](http://lloydsheng.com/wp-content/uploads/2010/04/image4.png)

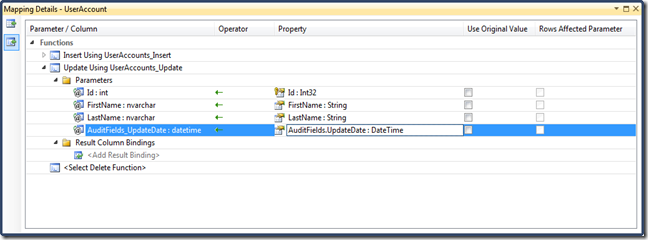
右击模型浏览器中的UserAccounts\_SelectById 存储过程选择Add Function Import，然后选择UserAccounts\_Select\_Result 类型作为该存储过程的返回值类型，点击OK。

下面是在UserAccount中使用插入，更新和删除的存储过程，右击设计器中的UserAccount实体选择Stored Procedure Mapping，会出现详细映射窗口。

点击第一行的<Select Insert Function>，在下拉列表中选择UserAccounts\_Insert ，在格子中会显示参数列表。我们必须进行实体属性到存储过程参数的映射。由于存储过程会返回新创建记录的Id，必须选择该Id赋值到哪个属性。在Result Column Bindings节点下的<Add Result Binding>输入Id，这里返回的Id是存储过程中“SELECT SCOPE\_IDENTITY() AS Id”，下面是绑定完成后的结果：

[](http://lloydsheng.com/wp-content/uploads/2010/04/image5.png)

映射更 新的存储过程，将<Select Update Function>替换为UserAccounts\_Update存储过程。其他的属性自动映射完成，只有AuditFields\_UpdateDate需要自动映射为AuditFields.UpdateDate属性。

[](http://lloydsheng.com/wp-content/uploads/2010/04/image6.png)

最后是映射Delete存储过程，将<Select Delete Function>替换为UserAccounts\_Delete存储过程，Id参数映射到和实体的Id属性。

不要忘了保存上面的操作结果，OrdersDBContainer 现在就能使用那些存储过程了，从而进行UserAccount记录的CRUD操作。

**步骤3：创建Web表单**

下面将在程序中创建一个表单，用来管理UserAccount数据。

1.在项目文件上右键，选择Add->New Item..

2.选择Web Form模板，将名字改为UsersSP.aspx，点“Add”。

3.在UserSP.aspx的div之间添加如下代码：

<table>

<tr>

<td>Select A User:</td>

<td><asp:DropDownList runat=server ID="ddlUsers" AutoPostBack="True">

</asp:DropDownList> </td>

</tr>

<tr>

<td>First Name:</td>

<td><asp:TextBox runat="server" ID="txtFirstName"></asp:TextBox></td>

</tr>

<tr>

<td>Last Name:</td>

<td><asp:TextBox runat="server" ID="txtLastName"></asp:TextBox></td>

</tr>

<tr>

<td>Inserted:</td>

<td><asp:Label runat="server" ID="lblInserted"></asp:Label> </td>

</tr>

<tr>

<td>Updated:</td>

<td><asp:Label runat="server" ID="lblUpdated"></asp:Label> </td>

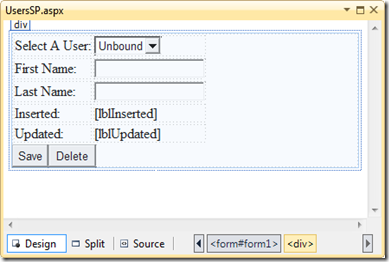
</tr>

</table>

<asp:Button runat=server ID="btnSave" Text="Save" />

<asp:Button ID="btnDelete" runat="server" Text="Delete" />

这里采用的table的对form进行了简单的布局。转到Design视图你会看到表单的样子如下：

[](http://lloydsheng.com/wp-content/uploads/2010/04/image7.png)

**步骤4：将数据加载到Drop Down List中**

要完成的是在页面加载时，将UserAccount的Name和Id数据加载到Drop Down List中。当选择特定项时，加载更加详细的信息。

1.双击Degsin视图（F7），在后台代码中添加Page\_Load 事件。

2.处理Page\_Load 事件的代码如下：

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

LoadUserDropDownList();

}

}

3.Page\_Load方法中的LoadUserDropDownList方法代码如下：

private void LoadUserDropDownList()

{

ddlUsers.DataSource = from u in db.UserAccounts\_SelectAll()

orderby u.LastName

select new { Name = u.LastName + ", " + u.FirstName,

Id = u.Id };

ddlUsers.DataTextField = "Name";

ddlUsers.DataValueField = "Id";

ddlUsers.DataBind();

ddlUsers.Items.Insert(0, new ListItem("Create New User", ""));

}

}

注意Linq的from语句，它调用OrderDBContainer的UserAccounts\_SelectAll方法，这个方法将会执行存储过程。

DataTextField属性设置为Name，DataValueField 设置为Id，这些都是在Linq查询中创建的。设置完成后，就是绑定了。绑定时，才真正开始调用数据库查询的操作。最后给Drop Down List添加一项“Crate New User.”，这项是用来区分更新和添加操作的。

现在数据库中还没有任何数据，Drop Down List中只有"Crete New User"一项。

**步骤5：添加和更新数据**

下面将向你介绍如何添加和更新表中的数据。

1.转到设计视图，双击Save按钮，创建该按钮的点击事件。

2.处理点击事件的代码如下：

using (OrderDBContainer db = new OrderDBContainer())

{

UserAccount userAccount = new UserAccount();

userAccount.FirstName = txtFirstName.Text;

userAccount.LastName = txtLastName.Text;

userAccount.AuditFields.UpdateDate = DateTime.Now;

if (ddlUsers.SelectedItem.Value == "")

{

//Adding

userAccount.AuditFields.InsertDate = DateTime.Now;

db.UserAccounts.AddObject(userAccount);

}

else

{

//Updating

userAccount.Id = Convert.ToInt32(ddlUsers.SelectedValue);

userAccount.AuditFields.InsertDate = Convert.ToDateTime(lblInserted.Text);

db.UserAccounts.Attach(userAccount);

db.ObjectStateManager.ChangeObjectState(userAccount, System.Data.EntityState.Modified);

}

db.SaveChanges();

lblInserted.Text = userAccount.AuditFields.InsertDate.ToString();

lblUpdated.Text = userAccount.AuditFields.UpdateDate.ToString();

//Reload the drop down list

LoadUserDropDownList();

//Select the one the user just saved.

ddlUsers.Items.FindByValue(userAccount.Id.ToString()).Selected = true;

}

}

代码首先创建OrderDBContainer对象，再创建UserAccount对象，用输入的值填充UserAccount对象属性。更新日期用系统当前时间，接着判断是更新操作还是添加操作了。最后就是更新Drop Down List的值并选中刚刚操作的UserAccout。这里的db.SaveChanges()最后实际上是在数据库中执行添加或更新语句。想数据库添加数据时，打开SQL Profiler会看到Insert存储过程被调用。

exec [dbo].[UserAccounts\_Insert]

@FirstName=N'Lloyd',

@LastName=N'Sheng',

@AuditFields\_InsertDate='2010-04-26 18:14:42.4564241',

@AuditFields\_UpdateDate='2010-04-26 18:14:42.4564241'

**步骤6:查询数据**

下面是实现当用户选择某一个Drop Down List项时，显示详细信息的功能。

1.双击视图设计器中的Drop Down List，这时会创建Drop Down List的SelectedIndexChanged方法。

2.编写SelectedIndexChanged方法的代码如下：

if (ddlUsers.SelectedValue == "")

{

txtFirstName.Text = "";

txtLastName.Text = "";

lblInserted.Text = "";

lblUpdated.Text = "";

}

else

{

//Get the user from the DB

using (OrderDBContainer db = new OrderDBContainer())

{

int userAccountId = Convert.ToInt32(ddlUsers.SelectedValue);

var userAccounts = from u in db.UserAccounts\_SelectById(userAccountId)

select u;

txtFirstName.Text = "";

txtLastName.Text = "";

lblInserted.Text = "";

lblUpdated.Text = "";

foreach (UserAccounts\_Select\_Result userAccount in userAccounts)

{

txtFirstName.Text = userAccount.FirstName;

txtLastName.Text = userAccount.LastName;

lblInserted.Text = userAccount.AuditFields\_InsertDate.ToString();

lblUpdated.Text = userAccount.AuditFields\_UpdateDate.ToString();

}

}

}

代码根据Drop Down List选择的Id，调用UserAccounts\_SelectById方法从数据库中查询一条数据并且显示出来。

**步骤7：删除数据**

最后就是删除数据的功能了。 1.转到视图设计器，双击“Delete”按钮。 2.添加如下代码：

if using (OrderDBContainer db = new OrderDBContainer())

{

if (ddlUsers.SelectedItem.Value != "")

{

UserAccount userAccount = new UserAccount();

userAccount.Id = Convert.ToInt32(ddlUsers.SelectedValue);

db.UserAccounts.Attach(userAccount);

db.ObjectStateManager.ChangeObjectState(userAccount,

System.Data.EntityState.Deleted);

db.SaveChanges();

LoadUserDropDownList();

txtFirstName.Text = "";

txtLastName.Text = "";

lblInserted.Text = "";

lblUpdated.Text = "";

}

}

代码首先创建了一个UserAccount对象，将它的Id设置为选中项的Id.然后将UserAccount附加到UserAccount集合中，设置它的状态为删除。调用SaveChanges操作，将该条数据删除，刷新Drop Down List的数据源，搞定！

**小结**

第二篇文章中告诉你如何用Entity Framework进行CRUD操作，本篇文章张告诉你如何用存储过程和Entity Framework进行CRUD操作。

在下一篇文章中我将向你介绍如何在三层结构中使用Entity Framework，希望你能喜欢。

项目代码：http://files.cnblogs.com/lloydsheng/OrderSystem3.zip  
本文永久链接：http://lloydsheng.com/2010/04/aspnet40-entityframework4-execute-stored-procedures-using-entity-framework4.html