

LINQ to SQL (Part 4 - Updating our Database)

Over the last few weeks I've been writing a series of blog posts that cover LINQ to SQL. LINQ to SQL is a built-in O/RM (object relational mapper) that ships in the .NET Framework 3.5 release, and which enables you to easily model relational databases using .NET classes. You can use LINQ expressions to query the database with them, as well as update/insert/delete data.

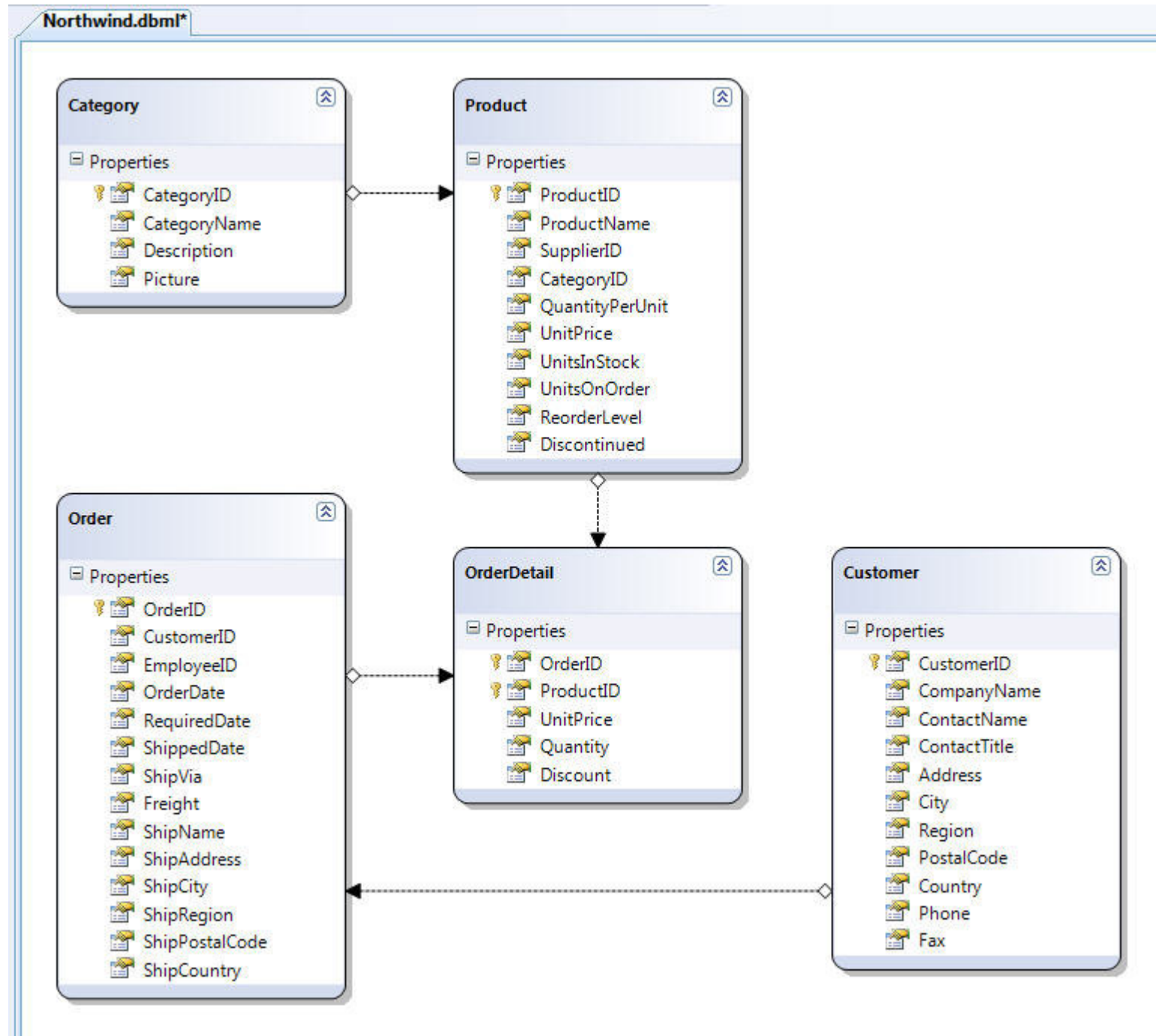
Below are the first three parts of my LINQ to SQL series:

- [Part 1: Introduction to LINQ to SQL](#)
- [Part 2: Defining our Data Model Classes](#)
- [Part 3: Querying our Database](#)

In today's blog post I'll cover how we can use the data model we created earlier, and use it to update, insert, and delete data. I'll also show how we can cleanly integrate business rules and custom validation logic with our data model.

Northwind Database Modeled using LINQ to SQL

In [Part 2](#) of this series I walked through how to create a LINQ to SQL class model using the LINQ to SQL designer that is built-into VS 2008. Below is a class model created for the Northwind sample database and which I'll be using in this blog post:



When we designed our data model using the LINQ to SQL data designer above we defined five data model classes: Product, Category, Customer, Order and OrderDetail. The properties of each class map to the columns of a corresponding table in the database. Each instance of a class entity represents a row within the database table.

When we defined our data model, the LINQ to SQL designer also created a custom DataContext class that provides the main conduit by which we'll query our database and apply updates/changes. In the example data model we defined above this class was named "NorthwindDataContext". The NorthwindDataContext class has properties that represent each Table we modeled within the database (specifically: Products, Categories, Customers, Orders, OrderDetails).

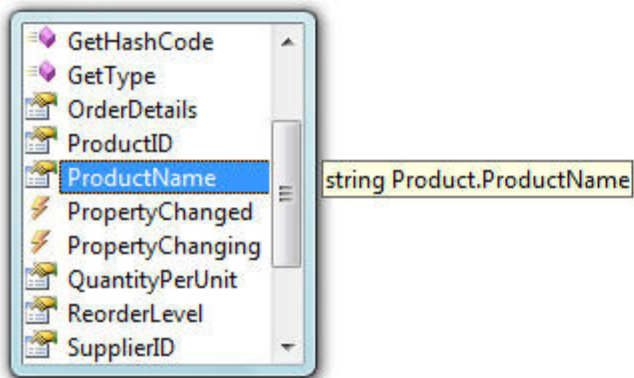
As I covered in [Part 3](#) of this blog series, we can easily use LINQ syntax expressions to query and retrieve data from our database using this NorthwindDataContext class. LINQ to SQL will then automatically translate these LINQ query expressions to the appropriate SQL code to execute at runtime.

For example, I could write the below LINQ expression to retrieve a single Product object by searching on the Product name:

```
NorthwindDataContext northwind = new NorthwindDataContext();

Product myProduct = northwind.Products.Single(p => p.ProductName == "Chai");

myProduct.
```



I could then write the LINQ query expression below to retrieve all products from the database that haven't yet had an order placed for them, and which also cost more than \$100:

```
NorthwindDataContext northwind = new NorthwindDataContext();

var products = from p in northwind.Products
               where p.OrderDetails.Count == 0 && p.UnitPrice > 100
               select p;
```

Note above how I am using the "OrderDetails" association for each product as part of the query to only retrieve those products that have not had any orders placed for them.

Change Tracking and DataContext.SubmitChanges()

When we perform queries and retrieve objects like the product instances above, LINQ to SQL will by default keep track of any changes or updates we later make to these objects. We can make any number of queries and changes we want using a LINQ to SQL DataContext, and these changes will all be tracked together.

Note: LINQ to SQL change tracking happens on the consuming caller side - and not in the database. This means that you are not consuming any database resources when using it, nor do you need to change/install anything in the database to enable it.

After making the changes we want to the objects we've retrieved from LINQ to SQL, we can then optionally call the "SubmitChanges()" method on our DataContext to apply the changes back to the database. This will cause LINQ to SQL to dynamically calculate and execute the appropriate SQL code to update the database.

For example, I could write the below code to update the price and # of units in stock of the "Chai" product in the database:

```
NorthwindDataContext northwind = new NorthwindDataContext();

Product myProduct = northwind.Products.Single(p => p.ProductName == "Chai");

myProduct.UnitPrice = 2;
myProduct.UnitsInStock = 4;

northwind.SubmitChanges();
```

When I call northwind.SubmitChanges() above, LINQ to SQL will dynamically construct and execute a SQL "UPDATE" statement that will update the two product property values we modified above.

I could then write the below code to loop over unpopular, expensive products and set the "ReorderLevel" property of them to zero:

```
NorthwindDataContext northwind = new NorthwindDataContext();

var expensiveUnpopularProducts = from p in northwind.Products
                                 where p.OrderDetails.Count == 0 && p.UnitPrice > 100
                                 select p;

foreach (Product product in expensiveUnpopularProducts) {
    product.ReorderLevel = 0;
}

northwind.SubmitChanges();
```

When I call northwind.SubmitChanges() above, LINQ to SQL will calculate and execute an appropriate set of UPDATE statements to modify the products who had their ReorderLevel property changed.

Note that if a Product's property values weren't changed by the property assignments above, then the object would not be considered changed and LINQ to SQL would therefore not execute an update for that product back to the database. For example - if the "Chai" product's unitprice was already \$2 and the number of units in stock was 4, then calling SubmitChanges() would not cause any database update statements to execute. Likewise, only those products in the second example whose ReorderLevel was not already 0 would be updated when the SubmitChanges() method was called.

Insert and Delete Examples

In addition to updating existing rows in the database, LINQ to SQL obviously also enables you to insert and delete data. You can accomplish this by adding/removing data objects from the DataContext's table collections, and by then calling the SubmitChanges() method. LINQ to SQL will keep track of these additions/removals, and automatically execute the appropriate SQL INSERT or DELETE statements when SubmitChanges() is invoked.

Inserting a New Product

I can add a new product to my database by creating a new "Product" class instance, setting its properties, and then by adding it to my DataContext's "Products" collection:


```
NorthwindDataContext northwind = new NorthwindDataContext();

Product myProduct = new Product();
myProduct.ProductName = "Scott's Special Product";
myProduct.UnitPrice = 999;
myProduct.UnitsInStock = 1;
myProduct.CategoryID = 1;

northwind.Products.Add(myProduct);

northwind.SubmitChanges();
```

When we call "SubmitChanges()" above a new row will be created in our products table.

Deleting Products

Just as I can express that I want to add a new Product to the database by adding a Product object into the DataContext's Products collection, I can likewise express that I want to delete a product from a database by removing it from the DataContext's Products collection:

```
NorthwindDataContext northwind = new NorthwindDataContext();

var lameProducts = from p in northwind.Products
                    where p.OrderDetails.Count > 0 && p.Discontinued == true
                    select p;

northwind.Products.RemoveAll(lameProducts);

northwind.SubmitChanges();
```

Note above how I'm retrieving a sequence of discontinued products that no one has ever ordered using a LINQ query, and then passing it to the RemoveAll() method on my DataContext's "Products" collection. When we call "SubmitChanges()" above all of these Product rows will be deleted from our products table.

Updates across Relationships

What makes O/R mappers like LINQ to SQL extremely flexible is that they enable us to easily model cross-table relationships across our data model. For example, I can model each Product to be in a Category, each Order to contain OrderDetails for line-items, associate each OrderDetail line-item with a Product, and have each Customer contain an associated set of Orders. I covered how to construct and model these relationships in [Part 2](#) of this blog series.

LINQ to SQL enables me to take advantage of these relationships for both querying and updating my data. For example, I could write the below code to create a new Product and associate it with an existing "Beverages" category in my database like so:

```
NorthwindDataContext northwind = new NorthwindDataContext();

// Retrieve beverages category
Category beverages = northwind.Categories.Single(c => c.CategoryName == "Beverages");

// Create new Product
Product myProduct = new Product();
myProduct.ProductName = "Scott's Special Product";
myProduct.UnitPrice = 999;
myProduct.UnitsInStock = 1;

// Associate product with beverage category
beverages.Products.Add(myProduct);

// Update database
northwind.SubmitChanges();
```

Note above how I'm adding the Product object into the Category's Products collection. This will indicate that there is a relationship between the two objects, and cause LINQ to SQL to automatically maintain the foreign-key/primary key relationship between the two when I call "SubmitChanges()".

For another example of how LINQ to SQL can help manage cross-table relationships for us and help clean up our code, let's look at an example below where I'm creating a new Order for an existing customer. After setting the required ship date and freight costs for the order, I then create two order line-item objects that point to the products the customer is ordering. I then associate the order with the customer, and update the database with all of the changes.

```

NorthwindDataContext northwind = new NorthwindDataContext();

// Retrieve product details
Product chai = northwind.Products.Single(p => p.ProductName == "Chai");
Product tofu = northwind.Products.Single(p => p.ProductName == "Tofu");

// Create new Order and Order line items for products
Order myOrder = new Order();
myOrder.OrderDate = DateTime.Now;
myOrder.RequiredDate = DateTime.Now.AddDays(1);
myOrder.Freight = 34;

OrderDetail myItem1 = new OrderDetail();
myItem1.Product = chai;
myItem1.Quantity = 23;

OrderDetail myItem2 = new OrderDetail();
myItem2.Product = tofu;
myItem2.Quantity = 3;

// Associate new order and order line items together
myOrder.OrderDetails.Add(myItem1);
myOrder.OrderDetails.Add(myItem2);

// Retrieve customer details
Customer myCustomer = northwind.Customers.Single(c => c.CompanyName == "B's Beverages");

// Add the order to the customer's Orders collection
myCustomer.Orders.Add(myOrder);

// Update database
northwind.SubmitChanges();

```

As you can see, the programming model for performing all of this work is extremely clean and object oriented.

Transactions

A transaction is a service provided by a database (or other resource manager) to guarantee that a series of individual actions occur atomically - meaning either they all succeed or they all don't, and if they don't then they are all automatically undone before anything else is allowed to happen.

When you call SubmitChanges() on your DataContext, the updates will always be wrapped in a Transaction. This means that your database will never be in an inconsistent state if you perform multiple changes - either all of the changes you've made on your DataContext are saved, or none of them are.

If no transaction is already in scope, the LINQ to SQL DataContext object will automatically start a database transaction to guard updates when you call SubmitChanges(). Alternatively, LINQ to SQL also enables you to explicitly define and use your own TransactionScope object (a feature introduced in .NET 2.0). This makes it easier to integrate LINQ to SQL code with existing data access code you already have. It also means that you can enlist non-database resources into the transaction - for example: you could send off a MSMQ message, update the file-system (using the new transactional file-system support), etc - and scope all of these work items in the same transaction that you use to update your database with LINQ to SQL.

Validation and Business Logic

One of the important things developers need to think about when working with data is how to incorporate validation and business rule logic. Thankfully LINQ to SQL supports a variety of ways for developers to cleanly integrate this with their data models.

LINQ to SQL enables you to add this validation logic once - and then have it be honored regardless of where/how the data model you've created is used. This avoids you having to repeat logic in multiple places, and leads to a much more maintainable and clean data model.

Schema Validation Support

When you define your data model classes using the LINQ to SQL designer in VS 2008, they will by default be annotated with some validation rules inferred from the schema of the tables in the database.

The datatypes of the properties in the data model classes will match the datatypes of the database schema. This means you will get compile errors if you attempt to assign a boolean to a decimal value, or if you attempt to implicitly convert numeric types incorrectly.

If a column in the database is marked as being nullable, then the corresponding property in the data model class created by the LINQ to SQL designer will be a nullable type. Columns not marked as nullable will automatically raise exceptions if you attempt to persist an instance with a null value. LINQ to SQL will likewise ensure that identity/unique column values in the database are correctly honored.

You can obviously use the LINQ to SQL designer to override these default schema driven validation settings if you want - but by default you get them automatically and don't have to take any additional steps to enable them. LINQ to SQL also automatically handles escaping SQL values for you - so you don't need to worry about SQL injection attacks when using it.

Custom Property Validation Support

Schema driven datatype validation is useful as a first step, but usually isn't enough for real-world scenarios.

Consider for example a scenario with our Northwind database where we have a "Phone" property on the "Customer" class which is defined in the database as an nvarchar. Developers using LINQ to SQL could write code like below to update it using a valid telephone number:

```

NorthwindDataContext northwind = new NorthwindDataContext();

Customer myCustomer = northwind.Customers.Single(c => c.CompanyName == "B's Beverages");
myCustomer.Phone = "425-703-8072";

northwind.SubmitChanges();

```

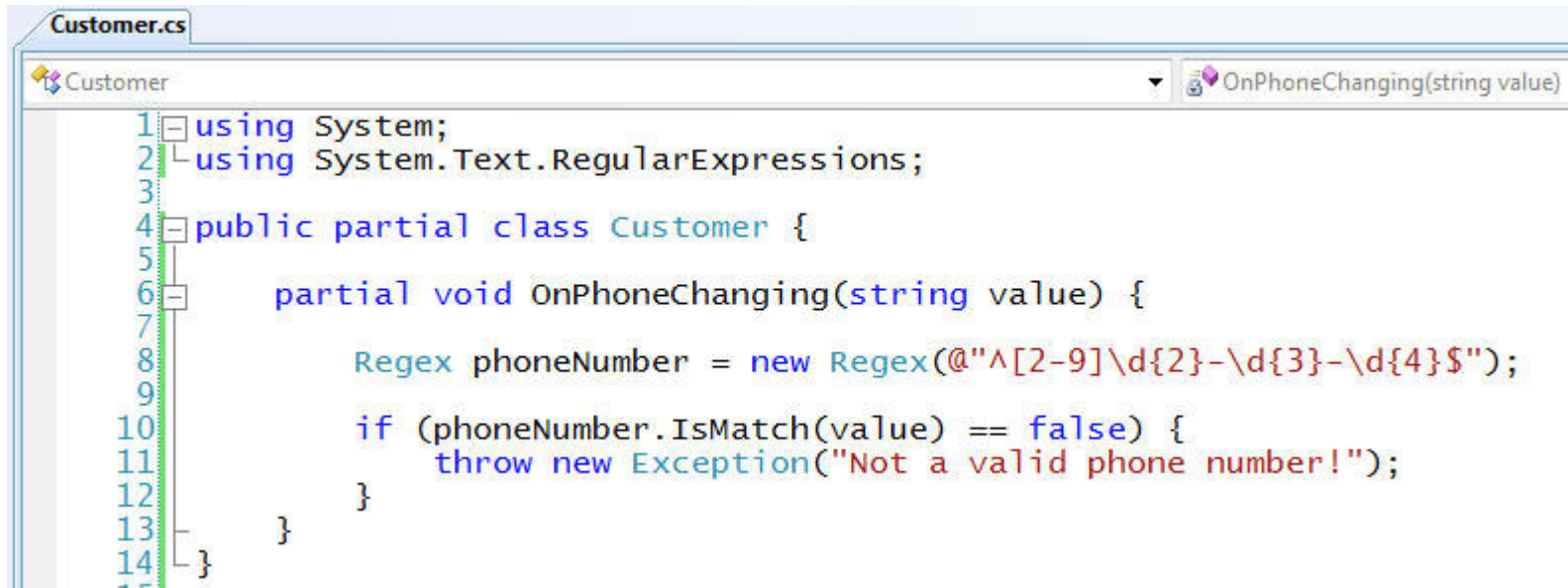
The challenge that we will run into with our application, however, is that the below code is also legal from a pure SQL schema perspective (because it is still a string even though it is not a valid phone number):


```
NorthwindDataContext northwind = new NorthwindDataContext();

Customer myCustomer = northwind.Customers.Single(c => c.CompanyName == "B's Beverages");
myCustomer.Phone = "absdfdsfdfsfd";

northwind.SubmitChanges();
```

To prevent bogus phone numbers from being added into our database, we can add a custom property validation rule to our Customer data model class. Adding a rule to validate phone numbers using this feature is really easy. All we need to-do is to add a new partial class to our project that defines the method below:



```
Customer.cs
Customer
1 using System;
2 using System.Text.RegularExpressions;
3
4 public partial class Customer {
5
6     partial void OnPhoneChanging(string value) {
7
8         Regex phoneNumber = new Regex(@"^[2-9]\d{2}-\d{3}-\d{4}$");
9
10        if (phoneNumber.IsMatch(value) == false) {
11            throw new Exception("Not a valid phone number!");
12        }
13    }
14 }
```

The code above takes advantage of two characteristics of LINQ to SQL:

- 1) All classes created by the LINQ to SQL designer are declared as "partial" classes - which means that developers can easily add additional methods, properties, and events to them (and have them live in separate files). This makes it very easy to augment the data model classes and DataContext classes created by the LINQ to SQL designer with validation rules and additional custom helper methods that you define. No configuration or code wire-up is required.
- 2) LINQ to SQL exposes a number of custom extensibility points in its data model and DataContext classes that you can use to add validation logic before and after things take place. Many of these extensibility points utilize a new language feature called "partial methods" that is being introduced with VB and C# in VS 2008 Beta2. Wes Dyer from the C# team has a good explanation of how partial methods works in this blog post [here](#).

In my validation example above, I'm using the OnPhoneChanging partial method that is executed anytime someone programmatically sets the "Phone" property on a Customer object. I can use this method to validate the input however I want (in this case I'm using a regular expression). If everything passes successfully, I just return from the method and LINQ to SQL will assume that the value is valid. If there are any issues with the value, I can raise an exception within the validation method - which will prevent the assignment from taking place.

Custom Entity Object Validation Support

Property level validation as used in the scenario above is very useful for validating individual properties on a data model class. Sometimes, though, you want/need to validate multiple property values on an object against each other.

Consider for example a scenario with an Order object where you set both the "OrderDate" and the "RequiredDate" properties:

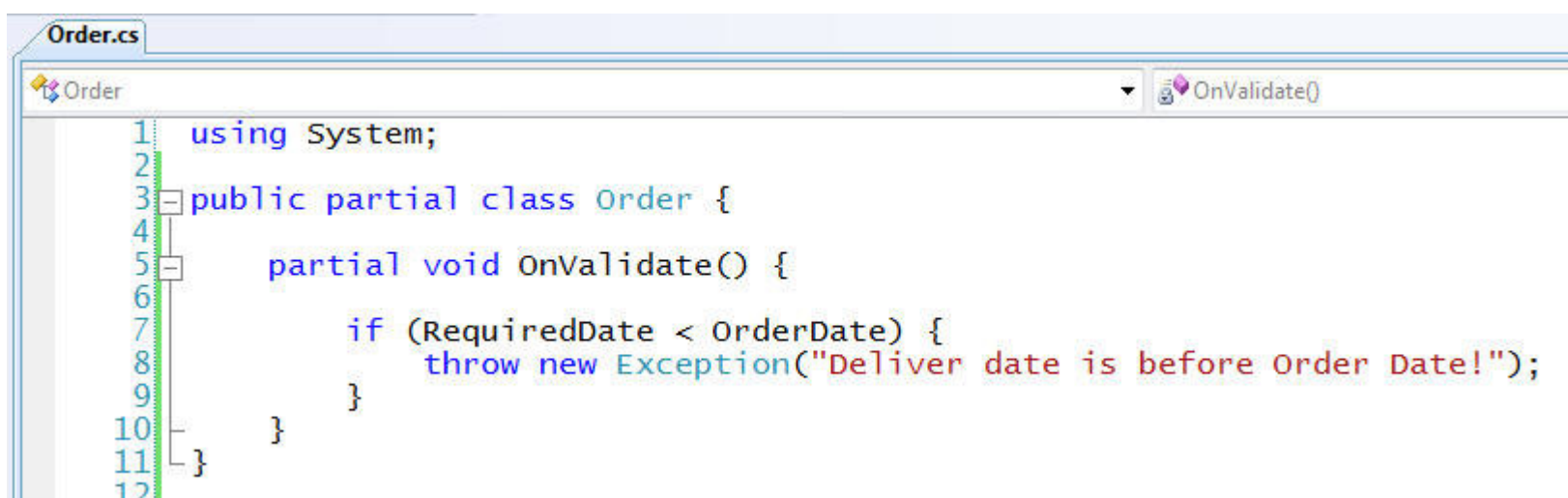
```
NorthwindDataContext northwind = new NorthwindDataContext();

Order myOrder = new Order();
myOrder.OrderDate = DateTime.Now;
myOrder.RequiredDate = DateTime.Now.AddDays(-1);

northwind.Orders.Add(myOrder);
northwind.SubmitChanges();
```

The above code is legal from a pure SQL database perspective - even though it makes absolutely no sense for the required delivery date of the new order to be entered as yesterday.

The good news is that LINQ to SQL in Beta2 makes it easy for us to add custom entity level validation rules to guard against mistakes like this from happening. We can add a partial class for our "Order" entity and implement the OnValidate() partial method that will be invoked prior to the entity's values being persisted into the database. Within this validation method we can then access and validate all of the data model class properties:



```
Order.cs
Order
1 using System;
2
3 public partial class Order {
4
5     partial void OnValidate() {
6
7         if (RequiredDate < OrderDate) {
8             throw new Exception("Deliver date is before Order Date!");
9         }
10    }
11 }
12
```

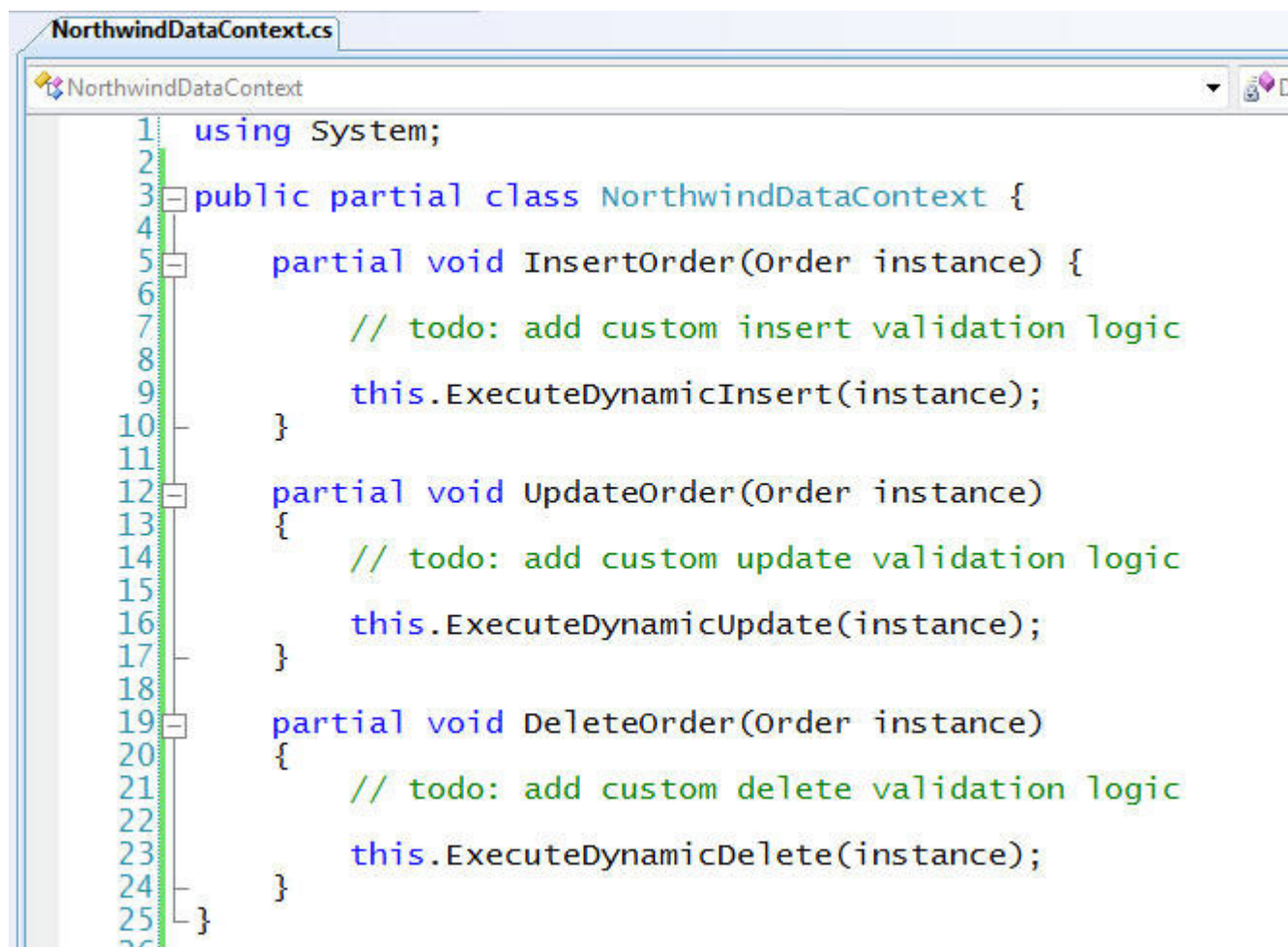
Within this validation method I can check any of the entity's property values (and even obtain read-only access to its associated objects), and raise an exception as needed if the values are incorrect. Any exceptions raised from the OnValidate() method will abort any changes from being persisted in the database, and rollback all other changes in the transaction.

Custom Entity Insert/Update/Delete Method Validation

There are times when you want to add validation logic that is specific to insert, update or delete scenarios. LINQ to SQL in Beta2 enables this by allowing you to add a partial class to

extend your DataContext class and then implement partial methods to customize the Insert, Update and Delete logic for your data model entities. These methods will be called automatically when you invoke SubmitChanges() on your DataContext.

You can add appropriate validation logic within these methods - and if it passes then tell LINQ to SQL to continue with persisting the relevant changes to the database (by calling the DataContext's "ExecuteDynamicXYZ" method):



```
1 using System;
2
3 public partial class NorthwindDataContext {
4
5     partial void InsertOrder(Order instance) {
6         // todo: add custom insert validation logic
7         this.ExecuteDynamicInsert(instance);
8     }
9
10    partial void UpdateOrder(Order instance)
11    {
12        // todo: add custom update validation logic
13        this.ExecuteDynamicUpdate(instance);
14    }
15
16    partial void DeleteOrder(Order instance)
17    {
18        // todo: add custom delete validation logic
19        this.ExecuteDynamicDelete(instance);
20    }
21 }
22
23
24
25
26
```

What is nice about adding the above methods is that the appropriate ones are automatically invoked regardless of the scenario logic that caused the data objects to be created/updated/deleted. For example, consider a simple scenario where we create a new Order and associate it with an existing Customer:

```
NorthwindDataContext northwind = new NorthwindDataContext();

// Retrieve customer
Customer myCustomer = northwind.Customers.Single(c => c.CompanyName == "Microsoft");

// Create new Order
Order myOrder = new Order();
myOrder.OrderDate = DateTime.Now;
myOrder.ShipAddress = "One Microsoft Way";
myOrder.ShipPostalCode = "98052";

// Associate order with customer
myCustomer.Orders.Add(myOrder);

// Update changes
northwind.SubmitChanges();
```

When we call northwind.SubmitChanges() above, LINQ to SQL will determine that it needs to persist a new Order object, and our "InsertOrder" partial method will automatically be invoked.

Advanced: Looking at the Entire Change List for the Transaction

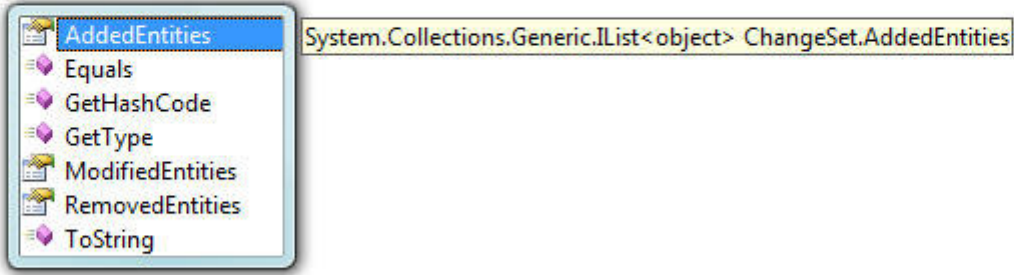
There are times when adding validation logic can't be done purely by looking at individual insert/update/delete operations - and instead you want to be able to look at the entire change list of operations that are occurring for a transaction.

Starting with Beta2 of .NET 3.5, LINQ to SQL now enables you to get access to this change list by calling the public DataContext.GetChangeList() method. This will return back a ChangeList object that exposes collections of each addition, removal and modification that has been made.

One approach you can optionally employ for advanced scenarios is to sub-class the DataContext class and override its SubmitChange() method. You can then retrieve the ChangeList() for the update operation and perform any custom validation you want prior to executing it:


```

public class MyNorthwindDataContext : NorthwindDataContext
{
    public override void SubmitChanges(System.Data.Linq.ConflictMode failureMode)
    {
        ChangeSet changes = this.GetChangeSet();

        changes.
        
        base.SubmitChanges(failureMode);
    }
}

```

The above scenario is a somewhat advanced one - but it is nice to know that you always have the ability to drop-down and take advantage of it if needed.

Handling Simultaneous Changes with Optimistic Concurrency

One of the things that developers need to think about in multi-user database systems is how to handle simultaneous updates of the same data in the database. For example, assume two users retrieve a product object within an application, and one of the users changes the ReorderLevel to 0 while the other changes it to 1. If both users then attempt to save the product back to the database, the developer needs to decide how to handle the change conflicts.

One approach is to just "let the last writer win" - which means that the first user's submitted value will be lost without the end-users realizing it. This is usually considered a poor (and incorrect) application experience.

Another approach which LINQ to SQL supports is to use an optimistic concurrency model - where LINQ to SQL will automatically detect if the original values in the database have been updated by someone else prior to the new values being persisted. LINQ to SQL can then provide a conflict list of changed values to the developer and enable them to either reconcile the differences or provide the end-user of the application with UI to indicate what they want to-do.

I'll cover how to use optimistic concurrency with LINQ to SQL in a future blog post.

Using SPROCs or Custom SQL Logic for Insert/Update/Delete Scenarios

One of the questions that developers (and especially DBAs) who are used to writing SPROCs with custom SQL usually ask when seeing LINQ to SQL for the first time is - "but how can I have complete control of the underlying SQL that is executed?"

The good news is that LINQ to SQL has a pretty flexible model that enables developers to override the dynamic SQL that is automatically executed by LINQ to SQL, and instead call custom insert, update, delete SPROCs that they (or a DBA) define themselves.

What is really nice is that you can start off by defining your data model and have LINQ to SQL automatically handle the insert, update, delete SQL logic for you. You can then at a later point customize the data model to use your own custom SPROCs or SQL for updates - without having to change any of the application logic that is using your data model, nor would you have to change any of the validation or business rules logic supporting it (all of this stays the same). This provides a lot of flexibility in how you build your application.

I'll cover how to customize your data models to use SPROCs or custom SQL in a future blog post.


Summary

Hopefully the above post provides a good summary of how you can easily use LINQ to SQL to update your database, and cleanly integrate validation and business logic with your data models. I think you'll find that LINQ to SQL can dramatically improve your productivity when working with data, and enable you to write extremely clean object-oriented data access code.

In upcoming blog posts in this series I'll cover the new <asp:linqdatasource> control coming in .NET 3.5, and talk about how you can easily build data UI in ASP.NET that takes advantage of LINQ to SQL data models. I'll also cover some more specific LINQ to SQL programming concepts including optimistic concurrency, lazy and eager loading, table mapping inheritance, custom SQL/SPROC usage, and more.

Hope this helps,

Scott

Published Wednesday, July 11, 2007 10:00 AM by ScottGu 
 Filed under: [ASP.NET](#), [Visual Studio](#), [.NET](#), [LINQ](#), [Data](#), [SQL Server](#)

Comments

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 1:22 PM by [rodrigo](#)

Hi scott great articles but I don't think there is an example of how

to do a LINQ code that retrieves data from an outer join.

Could you provide a simple snippet to do that?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 1:30 PM by Stefan Jokull

Awesome article Scott.

One question. When inserting an object that has it's ID mapped to an Identity column, is the Id property on the object automatically updated after calling SubmitChanges() or would we have to fetch the newly created object from the datacontext?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 1:33 PM by Matt Griffore

Two words... very nice!!

I just hope it plays well with Oracle.

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 1:44 PM by John West

Scott,

Take the scenario below.

- 1) Load order from northwinddatacontext
- 2) Update order with values from input screen - update some of the detail lines as well
- 3) Call northwind.submitchanges - let's assume the order returns a validation error
- 4) Display returned error
- 5) Create logentryobject with error details
- 6) Add logentryobject to northwinddatacontext
- 7) Call northwind.submitchanges

Now, won't step 7 fail again, since the order object is still in the change list? Is there a way to remove it from the context? Can we add it back later? Will removing it also remove the updated order detail objects?

This is an issue I run into often with ORMs, such as nhibernate, that save all changes with a single method, instead of requiring an explicit save method for each object.

Thanks.

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 1:46 PM by John West

One more question: when is automatic updating of the data model going to be added? Currently, if I add a new field to the db, I have to manually make the changes. Can't there be a sync option?

Thanks again.

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 1:56 PM by lee

I like linq to sql. do we use ADO.NET entities or ling to sql in DAL

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 2:17 PM by Alex Osipov

Hi Scott - just wanted to clarify something.

In your example:

```
var products =  
  
from p in northwind.Products  
  
where p.OrderDetails.Count == 0 && p.unitPrice > 100  
  
select p;  
  
foreach (Product product in products) {  
  
product.ReorderLevel = 0;  
  
}  

```

Will this execute multiple UPDATE statements for each product primary key or a single UPDATE statement?

AO

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 2:21 PM by [ScottGu](#)

Hi Stefan,

>>> One question. When inserting an object that has it's ID mapped to an Identity column, is the Id property on the object automatically updated after calling SubmitChanges() or would we have to fetch the newly created object from the datacontext?

When you are inserting a new object, after you call SubmitChanges() on it the Id value of the identity column will automatically be updated. So no need to explicitly fetch it.

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 2:25 PM by [ScottGu](#)

Hi John,

If I understand your scenario correctly, you are looking to create a logentry for tracking purposes when errors occur - and want to avoid the initial validation error blocking it.

For this scenario I think the best approach might be to just instantiate a new DataContext instance. Using this you could add the log entry and persist it without worrying about the other DataContext the user is updating (and which currently has a validation error).

Does this answer your question?

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 2:28 PM by [ScottGu](#)

Hi Alex,

For the looping update scenario I did above this will create multiple update statements (one for each product I'm updating). Note that only those products that are changed will have update statements though (so if the ReorderLevel value was already 0 no update statement would occur).

I believe LINQ to SQL will actually send the update statements together batched to the database, though, so that it might only be one network trip to the database.

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 3:05 PM by [Josh Stodola](#)

Hey there Scott, thanks again for the intriguing LINQ posts.

I am gonna ask this one last time... will you be showing some love to VB programmers and offering code in this great series for both languages? I would *really* appreciate this.

Thanks!

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 3:09 PM by JoeBo

I've been checking this stuff out for a while now and i really like what i see. My question to you scott is; will linq replace the need for stored procedures in the future?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 3:12 PM by Ben Hayat

Hi Scott;

All your posts are great, but this one has topped them all :-)

Question for you: So far, there has not been any mention of how a remote client can work with database (in 3-tier model). i.e. Let's say the database resides in a company and some users have direct connection to database via network. In this case all the above examples work great. But, what about users who are in a different city, or traveling who only have access to internet and not direct connection. The next project That I have to work on, is a service application that none of the users have direct access to database via network. These are small companies who will use our software, but their data will reside in a hosting company.

How can we leverage this new model in a 3-tier environment?

Thank you in advance Scott!

..Ben

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 3:18 PM by [ScottGu](#)

Hi Josh,

I'm planning on updating all of these posts to VB in the future. I started off the LINQ to SQL series doing both languages, but found that it ended up being really hard to read the posts when everything was duplicated. I'll be creating separate copies that contain pure VB in the future and link off to all of them.

Thanks,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 3:22 PM by [ScottGu](#)

Hi JoeBo,

>>>> I've been checking this stuff out for a while now and i really like what i see. My question to you scott is; will linq replace the need for stored procedures in the future?

People use stored procedures for different reasons, so I think it is often hard to say whether using an OR/M implementation is absolutely better (it really depends on the exact scenario).

I do, though, think that using an OR/M implementation like LINQ to SQL can make you much more productive, and provides tremendous flexibility to your code. One nice thing about LINQ to SQL is that it supports using both dynamic SQL (that it calculates and executes for you) as well as the ability to use SPROC's that you define. This makes it easy to use a data model built using LINQ to SQL, and dyanmic SQL for most scenarios, and then optimize/customize certain scenarios using SPROC's if needed.

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 3:25 PM by [ScottGu](#)

Hi Ben,

>>>> Question for you: So far, there has not been any mention of how a remote client can work with database (in 3-tier model). i.e. Let's say the database resides in a company and some users have direct connection to database via network. In this case all the above examples work great. But, what about users who are in a different city, or traveling who only have access to internet and not direct connection. The next project That I have to work on, is a service application that none of the users have direct access to database via network. These are small companies who will use our software, but their data will reside in a hosting company.

You can send LINQ to SQL data model classes over the network via web-services or other remoting technologies. For disconnected traveling scenarios, though, one thing to consider would be to use either SQL Express or SQL CE on the local client and cache/sync data locally there. You could then use LINQ to SQL on the backend master server, and use LINQ to SQL locally on the client.

Today you have to write code to manage some of this disconnected sync and update logic yourself. In the future you'll see .NET have more built-in support for patterns like this, and built-in plumbing infrastrature that will really help with building those types of systems.

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 3:28 PM by [ScottGu](#)

Hi Rodrigo,

>>>> Hi scott great articles but I don't think there is an example of how to do a LINQ code that retrieves data from an outer join. Could you provide a simple snippet to do that?

I'll put this on the list to blog more about. :-) You can definitely do JOINS using LINQ and LINQ to SQL. For example:

```
var q =  
from s in db.Suppliers  
join c in db.Customers on s.City equals c.City  
select new {  
Supplier = s.CompanyName,  
Customer = c.CompanyName,  
City = c.City
```

};

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 3:33 PM by Chris Moseley

Hi Scott,

In a future post could you please cover creating dynamic queries with LINQ To Sql (as in 'order by "fieldName")'? I'm interested to know how to do this correctly.

Thanks,

Chris

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 3:45 PM by Martin Nyborg

I know that you are mister ASPX but are you going to write about remoting the data context? There should be a mini data context in the beta 2 (hope there is)

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 3:51 PM by roncansan

Scott,

How do LINQ compare against DataObjects.NET "www.x-tensive.com/Products/DataObjects.NET/"

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 4:13 PM by Joe Chung

Hey Scott, great post!

Question: your code sample has DataContext.GetChangeSet() while you refer to the method in your explanation as GetChangeList(). Which is it supposed to be?

[# LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 5:46 PM by DotNetKicks.com

You've been kicked (a good thing) - Trackback from DotNetKicks.com

[# New and Notable 177](#)

Wednesday, July 11, 2007 5:58 PM by [Sam Gentile](#)

Multithreading and Concurrency Software Transactional Memory Part IV - Thread-Bound Transactions Software

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 6:10 PM by John West

Using a new datacontext seems like one way to do it. However, is there a way to remove an object from the change list? And will child objects that have been changed be removed as well?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 6:11 PM by John West

How about my other question ;)? Is synchronization between the db and the model coming soon?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 6:29 PM by Mike

It looks better each time I read about it. I will try the next beta for sure.

One question: what's your view on validation in the model vs. validation using the ASP.NET validators? Usually we're writing our validation twice. In your phone number example you'd also put a RegularExpressionValidator on the textbox in the form. Do you have any ideas or plans to tackle this issue?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 6:32 PM by AzamSharp

Hi Scott,

How would you delete a product and then persist in the database?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 6:37 PM by Pete

Hey Scott,

Your articles are always spot-on and really perform a valuable function for the .NET community.

As I'm feeling pedantic, I have to pull you up on thowing System.Exception in you validation code. I know that these are only examples, but examples from such distinguished sources (although I really think you must have a ghostwriter...) can only contribute to the really bad exception-related code that's prolific in the .NET world. InvalidOperationException would be nice!

Pete. :-)

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 11, 2007 6:49 PM by Scott Roberts

Good info. Thanks for taking the time to write these up. But don't forget your promise to address the use of LINQ to SQL in an ASP.NET environment. :)

Also, it looks like this design is going to lead people into using multiple DataContext objects in order to manage updates. When working with multiple DataContext objects, it's fairly important to mention that since the DataContext is the one tracking changes on an object, if you move the object to a different DataContext you "lose" all the changes made to that object (i.e. the object appears "clean" to the new DataContext).

I would have preferred to see objects track their own changes and to have finer control over updates (i.e. myObject.Save()), but it appears MS has gone in a different direction. I've never worked in this type of design before (all objects updated at once) and it scares me a little. Has anyone worked with this type of ORM before? Did you like it or hate it?

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 11, 2007 6:50 PM by Pete

.... although, perhaps that should rather be an ArgumentException for the OnPhoneChanging method. It's tricky! Grrr... type-centric exception handling is too unproductive and costly to do properly. I wonder if there's a better way.

LINQ to SQL part 2

Wednesday, July 11, 2007 7:09 PM by [Tiernans Comms Closet](#)

Part 4 has been released . info on parts 1, 2 and 3 is here .

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 11, 2007 8:40 PM by zeeshan hirani

thks scott for a great article. now one question i have is how do i go about doing something after the update of an object say an order object. i have been using paul wilson ormapper which raises events like on creating oncreated, onupdating,onupdated,onmaterializing etc so onupated event is where i create an instance of my order history object and add it to the order history collection of order object so is there an event for onupdated that i can use to save other objects within the same transaction. one other constraints that i have faced with wilson is that in these kinds of events if u try to query for say other order objects, it does not work because the transaction is already issued and as a result table is locked so querying for other order objects does not work. i hope linq realizes this issue and uses the existing transtion object to query for theobjects

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 11, 2007 9:29 PM by Daniel Melo

Hi Scott,

Please consider the following scenario, where you retrieve one category and associates one new product to it :

Category beverages = northwind.Categories.Single(c => c.CategoryName == 'Beverages');

Product myProduct = new Product();

myProduct.ProductName = "Brand new product";

beverages.Products.Add(myProduct);

northwind.SubmitChanges();

If I am not missing something, this operation will trigger 2 network roundtrips to database :

a) One select operation to fetch the category "Beverages";

b) One insert to add the new product.

My question: If we already know the category primary key in advance, for example taking from query string, can we still add the new product, without fetching it from database?

Example:

Category beverages = new Category();

beverage.CategoryID = Request.QueryString["CategoryID"];

Product myProduct = new Product();

myProduct.ProductName = "Brand new product";

beverages.Products.Add(myProduct);

If it is possible, it would save me one round trip to fetch the category...

Thanks!

Daniel

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 11, 2007 9:52 PM by [ScottGu](#)

Hi John,

>>>>> Using a new datacontext seems like one way to do it. However, is there a way to remove an object from the change list? And will child objects that have been changed be removed as well?

You can remove items from the change-list, as well as optionally reset objects to their original state. So that would be another way to go if you wanted to. For the logging scenario, though, I think using a new DataContext is probably easiest.

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 11, 2007 9:55 PM by [ScottGu](#)

Hi Mike,

>>>>> One question: what's your view on validation in the model vs. validation using the ASP.NET validators? Usually we're writing our validation twice. In your phone number example you'd also put a RegularExpressionValidator on the textbox in the form. Do you have any ideas or plans to tackle this issue?

In general you definitely want to try and avoid duplicating validation logic. The nice thing about putting the validation logic in the data model is that you can author it once and know that regardless of the code-path that interacts with the data model the validation will execute.

Where things get a little trickier is with things like client-side JavaScript validation. For a certain set of validations (like a regular expression based email/telephone check) you could envision validation controls automatically basing their logic on the data model annotations. The current <asp:regularexpression> validator doesn't support this yet - but I suspect it is something we'll want to consider for the future.

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 11, 2007 9:56 PM by [ScottGu](#)

Hi Azam,

>>>> How would you delete a product and then persist in the database?

I have a delete sample above that shows using the RemoveAll() method to remove a set of products. There is also a Remove() method that you can use to pass in a single Product object. Once you call SubmitChanges() on the DataContext this will cause the deletion to be persisted in the database.

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 11, 2007 9:58 PM by ScottGu

Hi Pete,

>>>>>> although, perhaps that should rather be an ArgumentException for the OnPhoneChanging method. It's tricky! Grrr... type-centric exception handling is too unproductive and costly to do properly. I wonder if there's a better way.

Probably an ArgumentException would be the best way to go. One nice thing is that you can choose whichever type of exception you want to raise. For large projects I could envision people wanting to create their own custom exception hierarchy.

Thanks,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 11, 2007 10:12 PM by ScottGu

Hi Scott,

>>>>>> I would have preferred to see objects track their own changes and to have finer control over updates (i.e. myObject.Save()), but it appears MS has gone in a different direction. I've never worked in this type of design before (all objects updated at once) and it scares me a little. Has anyone worked with this type of ORM before? Did you like it or hate it?

One of the challanges when saving on a per object level is that it gets tricky when doing updates across multiple objects simultaneously.

For example, assume you create an Order and two OrderDetails that are associated with it. When you call save on the OrderDetail, should it fail if the Order hasn't already been persisted or should it auto-save it? If you call save on the Order should it automatically save its associated OrderDetails or not? These decisions end up getting even more complicated when you factor primary keys/foreign keys into the equation - since the order of the persistance can sometimes determine whether they work or fail.

Calling SubmitChanges() on the DataContext helps ensure that objects are kept in a consistent state, and you can make multiple changes to the model without having to worry about the order in which you persist the changes.

Note that you can call SubmitChanges() multiple time on a DataContext. So if you do want to perform multiple separate updates you can follow a pattern like below:

```
NorthwindDataContext northwind = new NorthwindDataContext();
```

```
// Add new object
```

```
northwind.SubmitChanges();
```

```
// Make more changes
```

```
northwind.SubmitChanges();
```

```
// More changes
```

```
northwind.SubmitChanges();
```

This enables you to separate our your updates into multiple different operations if you want more granularity.

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 11, 2007 10:45 PM by ScottGu

Hi Daniel,

>>>>>> My question: If we already know the category primary key in advance, for example taking from query string, can we still add the new product, without fetching it from database?

Yes - you can definitely go this route. The nice thing with LINQ to SQL is that you assign things using either the primary-key/foreign-key, or by setting object references across the objects.

Hope this helps!

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 11, 2007 11:08 PM by Jamison R.

I've been really hesitant to use type of OR/M technology, and over the last few weeks your articles on Linq to Sql have changed my mind. Now when I'm writing "traditional" ADO at work I keep thinking how much nicer it could be...

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, July 12, 2007 12:29 AM by Mohammad Azam

Hi Scott,

Thanks for the reply. But for the Delete/Remove to work we will first need to get the object from the database that we are going to delete. Is'nt this an extra fetch to the database?

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, July 12, 2007 12:29 AM by Vikram

Only one word for the article Great!

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, July 12, 2007 12:39 AM by Dennis

How is it possible to dynamically make the LINQ?

E.g. I have two boolean variables and I want to add some sql qualification depending on either or both being true.

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, July 12, 2007 1:57 AM by ScottGu

Hi Mohammad,

Deleting the object the way I showed will require a fetch from the database. But doing so ensures that all validation logic you've applied to your data model successfully executes prior to the deletion being committed.

Thanks,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Thursday, July 12, 2007 2:03 AM by [ScottGu](#)

Hi Dennis,

>>>> How is it possible to dynamically make the LINQ? E.g. I have two boolean variables and I want to add some sql qualification depending on either or both being true.

LINQ queries are evaluated lazily - which means that they don't actually execute until you request the data. One of the nice benefits of this is that you can compose queries nicely, and build queries from other queries very elegantly.

For example, you could write the below code to conditionally add a qualification to a query:

```
NorthwindDataContext northwind = new NorthwindDataContext();
```

```
var products = from p in northwind.Products
```

```
    select p;
```

```
if (somecondition == true)
```

```
{
```

```
    products = from p in products
```

```
        where p.Discontinued == true
```

```
    select p;
```

```
}
```

```
foreach(Product p in products)
```

```
{
```

```
    // do something
```

```
}
```

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Thursday, July 12, 2007 2:29 AM by [SoftMind](#)

Thanks Scott,

Great Going with LINQ tutorials.

By chance, are you aware of any good books currently available that explains LINQ from scratch with new features of 3.5.

Any idea of new books that should be coming out in near future. Can you recommend few UPCOMING books on LINQ.

Any plans on modified New Tutorials by Scott Mitchel based on LINQ.

Thanks

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Thursday, July 12, 2007 2:42 AM by [ScottGu](#)

Hi SoftMind,

Unfortunately I don't know of any LINQ books currently out there - although I expect several to appear in the months ahead.

Probably the best way to learn LINQ is to check out my language tutorial series as well as this LINQ to SQL series and then start playing with Beta2 of VS 2008 once it ships.

Here are a number of videos you can download and watch today about LINQ to SQL that you might find very useful: mtauly.com/.../9317.aspx

We'll definitely be updating our data tutorial series later this year to incorporate LINQ and LINQ to SQL as part of them.

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Thursday, July 12, 2007 2:59 AM by Dennis

How do you add caching?

E.g. if you do .Single(p => p.Id = 3) several hundred times per second.

How is invalidation handled on write? How is writing to the object handled, especially when is the cache updated with the new value?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Thursday, July 12, 2007 3:45 AM by Albert

He Scott, great article. Appreciate it. Keep up the good work.

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Thursday, July 12, 2007 4:23 AM by kangnoz

someone said LINQ to SQL is slower than SQL.

Is it real?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Thursday, July 12, 2007 8:59 AM by [Andy Barker](#)

Another good one, Scott! One thing I'm curious about is how one would use LINQ-to-SQL to create relationships that span multiple databases rather than multiple tables in the same database. Does the O/RM designer allow you to wire-up multiple database instances?

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, July 12, 2007 10:15 AM by André Cardoso

Hi Scott,

I have a question about the RemoveAll method and the corresponding SQL query: does it removes the products one by one (delete from ... where productId = x) or does it delete the set (delete from ... where condition)?

Thanks

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, July 12, 2007 11:00 AM by [ScottGu](#)

Hi Andy,

>>>> Another good one, Scott! One thing I'm curious about is how one would use LINQ-to-SQL to create relationships that span multiple databases rather than multiple tables in the same database. Does the O/RM designer allow you to wire-up multiple database instances?

Each DataContext maps back to a single database instance. However, you can create multiple DataContexts, with one pointing to each database.

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, July 12, 2007 11:06 AM by [ScottGu](#)

Hi kangnoz,

>>>>> someone said LINQ to SQL is slower than SQL. Is it real?

In general performance really depends on what you are doing. If you have extremely well written SQL queries, then in general you aren't going to find that O/RM implementations are faster (since they execute SQL under the covers themselves). However, in general I'd say you also won't find O/RM implementations to be much slower either - since they typically execute good SQL and employ perf best practices - something which I've found the majority of developers code often doesn't.

Rico has published some good performance comparisons of LINQ to SQL to raw ADO.NET DataReaders on this blog. You can read more about it here: blogs.msdn.com/.../linq-to-sql-rico-drops-the-other-shoe.aspx

In this test LINQ to SQL when using compiled queries was within 7% of using a raw firehose SqlDataReader when retrieving data, and was actually slightly faster than the hand-coded ADO.NET SQL for updates.

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, July 12, 2007 12:01 PM by Mohammad Azam

Hi Scott,

>>>Deleting the object the way I showed will require a fetch from the database. But doing so >>>ensures that all validation logic you've applied to your data model successfully executes >>>prior to the deletion being committed.

I think fetching the object before deleting it can cause performance problems. The reason is what if the object have multiple foreign keys and it retrieves all the data from the other table? What if the object is heavily populated with lots of fields?

Validation and Business Logic using LINQ to SQL

Thursday, July 12, 2007 12:19 PM by [Public Sector Developer Weblog](#)

I have spoken to quite a few customers about LINQ & LINQ to SQL over the last few months. Oddly enough,

Validation and Business Logic using LINQ to SQL

Thursday, July 12, 2007 12:43 PM by [Noticias externas](#)

I have spoken to quite a few customers about LINQ & LINQ to SQL over the last few months. Oddly enough

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, July 12, 2007 12:47 PM by Ryan Haney

Scott,

Most of this is just what I have been looking for in LINQ! Up until now, I had been using reflection to interrogate the internal change tracking mechanism for changed entities. Also, the validation events will allow me to clean up some of my existing code. It's nice to see some public APIs.

Thanks!

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, July 12, 2007 3:13 PM by JasonBSteele

Scott,

Great article again Scott, but.. I see validation examples but no real business logic.

In my comment to Part 3 I said:

"For obvious reasons examples of implementing business logic can be pretty trivial (e.g. is ToDate > FromDate). I think it would be of real value if something a bit more substantial could be shown that involves more than one entity. For example: Is this a new Order, and if so deduct quantities from the Stock table for each Order Detail checking that they do not fall below 0." which you thought was a good suggestion. So I had to laugh when the example you gave was of comparing two dates! ;)

In your validation example above you say "Within this validation method I can check any of the entity's property values (and even obtain read-only access to its associated objects)"

So it looks like implementing business logic that affects other entities can't be done in

OnValidate(). Can other entities be updated during InsertOrder? Is this where I would put business logic rather than just validation logic?

Also, can I easily look at the original values of properties, or even collections of related entities? (It was a shame that with strongly typed datasets you had to go back up to the Object typed DataRow ("FieldName", OriginalValue) to get to original values)

Thanks again for all the time you devote to communicating these ideas and answering our questions,

Jason

[**# LINQ to SQL \(Part 4 - Updating our Database\) - ScottGu's Blog**](#)

Thursday, July 12, 2007 4:23 PM by [LINQ to SQL \(Part 4 - Updating our Database\) - ScottGu's Blog](#)

Pingback from [LINQ to SQL \(Part 4 - Updating our Database\) - ScottGu's Blog](#)

[**# re: LINQ to SQL \(Part 4 - Updating our Database\)**](#)

Thursday, July 12, 2007 6:39 PM by André Cardoso

How does the RemoveAll statement gets translated to SQL? As a series of "delete from Products where ProductId = x" or as "delete from Products where ProductId in (initial query)" or as "delete from Products where ProductId in (x, y, z)"

Thanks for the great series on Linq.

André Cardoso

[**# re: LINQ to SQL \(Part 4 - Updating our Database\)**](#)

Thursday, July 12, 2007 9:45 PM by Steve

Thanks for this post very helpfull. Although when will beta 2 be shipping so we can try partial methods ? or can you give an example of OnPhoneChanging without partial method use for us still on beta 1 ?

Also in a future post could you please give us an example of how we can move Linq over the internet to say a Silverlight client or do you know of an existing sample that does this ? eg. have a silverlight client get data from linq on the server user changes data in client and sends back to server for update.

Thanks again,

Steve

[**# re: LINQ to SQL \(Part 4 - Updating our Database\)**](#)

Friday, July 13, 2007 4:32 AM by Dennis

How do you do caching? Is it possible to intercept calls to e.g. .Single and serve an item from cache? How do you then do invalidation of same?

[**# re: LINQ to SQL \(Part 4 - Updating our Database\)**](#)

Friday, July 13, 2007 4:51 AM by Paraag

Hello Scott,

Any Idea which tools/advantages Visual Web Developer will carry for LINQ, when compared with VS2008. I am talking about SQL Designer, OR/M facilities, etc.

Are there any important tools/advantages missing for LINQ in VWD...?

Will we have VWD- BETA 2 for .Net 3.5.

Pl. guide us.

[**# re: LINQ to SQL \(Part 4 - Updating our Database\)**](#)

Friday, July 13, 2007 11:51 AM by [Koistya `Navin](#)

Naming convention question: How should I name SQL Server 2005 Db tables, ex.: [Membership].[User] or [Membership].[Users] (with or without 's' at the end of each table's name)?

..with respect to LINQ to SQL db model designing.

[**# re: LINQ to SQL \(Part 4 - Updating our Database\)**](#)

Friday, July 13, 2007 12:12 PM by [ScottGu](#)

Hi Mohammad,

>>>> I think fetching the object before deleting it can cause performance problems. The reason is what if the object have multiple foreign keys and it retrieves all the data from the other table? What if the object is heavily populated with lots of fields?

LINQ to SQL supports the ability to lazily populate associations - which means that if the object you are deleting has foreign key relationships, you do not need to retrieve the other data from the other rows.

Hope this helps,

Scott

[**# re: LINQ to SQL \(Part 4 - Updating our Database\)**](#)

Friday, July 13, 2007 12:14 PM by [ScottGu](#)

Hi André,

>>>>> How does the RemoveAll statement gets translated to SQL? As a series of "delete from Products where ProductId = x" or as "delete from Products where ProductId in (initial query)" or as "delete from Products where ProductId in (x, y, z)"

The Remove statements will be executed as delete statements that remove each row (not as a filter on the actual property - but rather one that filters by productId).

Hope this helps,

Scott

[**# re: LINQ to SQL \(Part 4 - Updating our Database\)**](#)

Friday, July 13, 2007 12:14 PM by [Vikram](#)

Can we also have video of these stuff in action. I hope I am not asking for too much

[**# re: LINQ to SQL \(Part 4 - Updating our Database\)**](#)

Friday, July 13, 2007 12:16 PM by [ScottGu](#)

Hi Dennis,

>>>> How do you do caching? Is it possible to intercept calls to e.g. .Single and serve an item from cache? How do you then do invalidation of same?

By default, if you perform the same query twice on the same DataContext, the second call will automatically be served from the cache.

You could alternatively build a richer cache mechanism that took LINQ queries and then first resolved them against a local cache before accessing the remote database. This isn't built-in today - although the hooks are there to enable it. Hopefully someone will go ahead and implement it!

Thanks,
Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Friday, July 13, 2007 12:20 PM by [ScottGu](#)

Hi Steve,

>>>>> Thanks for this post very helpfull. Although when will beta 2 be shipping so we can try partial methods ? or can you give an example of OnPhoneChanging without partial method use for us still on beta 1 ?

Beta2 will be out shortly (next week or so) so it is probably best to just wait and use that. :-)

>>>>> Also in a future post could you please give us an example of how we can move Linq over the internet to say a Silverlight client or do you know of an existing sample that does this ? eg. have a silverlight client get data from linq on the server user changes data in client and sends back to server for update.

Yep - this is on the list to blog. One of the powerful things about Silverlight is that it supports LINQ on the client, which enables a bunch of really powerful scenarios (where you can use LINQ to SQL on the server to retrieve data - send it down to the client, and then use LINQ to Objects to locally query it, make updates, and then send it back).

Thanks,
Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Friday, July 13, 2007 12:23 PM by [ScottGu](#)

Hi JasonBSteele,

>>>>>> In your validation example above you say "Within this validation method I can check any of the entity's property values (and even obtain read-only access to its associated objects)" So it looks like implementing business logic that affects other entities can't be done in OnValidate(). Can other entities be updated during InsertOrder? Is this where I would put business logic rather than just validation logic?

You can access any of the other entity objects within the OnValidate and OnPropertyChanging events - you just want to avoid modifying them (which you really shouldn't do with validation anyway).

For adding additional logic to insert/change more entities, you can do this either within the InsertOrder() method, or by adding your own method to the DataContext that you call manually for inserts - and in which you encapsulate all of the creation/update logic.

Hope this helps,
Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Friday, July 13, 2007 12:25 PM by [ScottGu](#)

Hi Koistya `Navin,

>>>>> Naming convention question: How should I name SQL Server 2005 Db tables, ex.: [Membership].[User] or [Membership].[Users] (with or without 's' at the end of each table's name)? ..with respect to LINQ to SQL db model designing.

LINQ to SQL supports automatic "pluralization" within the LINQ to SQL designer. This means that you can name a table "Users", "Products", "Orders", etc - and the data model classes generated will automatically be named "User", "Product", "Order" (no s).

You can always override the names though within the LINQ to SQL designer, though, so if you don't like the default name you can easily change it to whatever you want.

Hope this helps,
Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Friday, July 13, 2007 12:27 PM by [ScottGu](#)

Hi Vikram,

>>>>>> Can we also have video of these stuff in action. I hope I am not asking for too much

We'll definitely be doing a lot of videos showing LINQ to SQL off. Here is an old one from January of this year of me doing a basic one: weblogs.asp.net/.../video-using-linq-with-asp-net-in-vs-orcas-part-1.aspx

Hope this helps,
Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Friday, July 13, 2007 12:28 PM by [ScottGu](#)

Hi Paraag,

>>>>>> Any Idea which tools/advantages Visual Web Developer will carry for LINQ, when compared with VS2008. I am talking about SQL Designer, OR/M facilities, etc. Are there any important tools/advantages missing for LINQ in VWD...?

All of the LINQ to SQL features - both the runtime as well as all of the designer/tool support - is available in the free Visual Web Developer Express edition. So you don't miss anything.

Hope this helps,
Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Friday, July 13, 2007 12:53 PM by [Josh Stodola](#)

Thanks Scott, I will be looking forward to the VB examples. I can totally understand how it would have been confusing to originally include both samples.
THANK YOU VERY MUCH!!

re: LINQ to SQL (Part 4 - Updating our Database)

Friday, July 13, 2007 12:59 PM by Ben Hayat

Hi Scott;

One of your answers prompted another question:

>>Yep - this is on the list to blog. One of the powerful things about Silverlight is that it supports LINQ on the client, which enables a bunch of really powerful scenarios (where you can use LINQ to SQL on the server to retrieve data - send it down to the client, and then use LINQ to Objects to locally query it, make updates, and then send it back).<<

The SilverLight (SilverBullet) is going to solve many current issues for developing and deploying thin clients. Currently, for my thin client, I have to worry about installing all the .Net (if client doesn't

have), all the DLLs and any third party DLL.

If I develop a SilverLight app using C# that uses some third party DLL, and when the consumer wants to run my silverlight app, does the SilverLight deployment system(engine), downloads and installs any of my DLLs that I'm using in my App.?

Thanks!

P.S. I think SilverLight is going to be a major direction for new developments...

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Friday, July 13, 2007 2:39 PM by JasonBSteele

Scott,

Thanks for the feedback on wgere to put business rules.

I also mentioned being able to get at original values:

>Also, can I easily look at the original values of properties, or >even collections of related entities? (It was a shame that with >stongly typed datasets you had to go back up to the Object typed >DataRow("FieldName", OriginalValue) to get to original values)

Will they be available for use within my business logic, or will they only be available in the scope of the OnPropertyChanging? Using your example above you could use this.phoneNumber to get the original value before it is changed, but ideally the original value would always be available.

On a different subject, I tried returning one of the Entity objects from a web service but it complained about not being able to serialise circular references. I can't find much ino on the Detach and Attach methods - should I be Detaching it first?

Many thanks,

Jason

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Friday, July 13, 2007 3:17 PM by [ScottGu](#)

Hi Ben,

>>>>> If I develop a SilverLight app using C# that uses some third party DLL, and when the consumer wants to run my silverlight app, does the SilverLight deployment system(engine), downloads and installs any of my DLLs that I'm using in my App.?

If you reference an assembly from a Silverlight application, it will automatically be downloaded and run by Silverlight. You don't need to copy/deploy it onto a client machine manually.

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Friday, July 13, 2007 4:00 PM by Kiran

Hi Scott,

I am not yet very familiar with LINQ to SQL, started to learn. My question is "are there any chances for vulnerabilities like SQL Injection because of using LINQ to SQL in web applications"?

Thanks

Kiran

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Friday, July 13, 2007 4:02 PM by [ScottGu](#)

Hi Kiran,

>>>>> "are there any chances for vulnerabilities like SQL Injection because of using LINQ to SQL in web applications"?

LINQ to SQL is typesafe and automatically encodes all parameters correctly. This protects you against SQL Injection attacks.

Thanks,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Friday, July 13, 2007 4:57 PM by Ben Hayat

>>If you reference an assembly from a Silverlight application, it will automatically be downloaded and run by Silverlight. You don't need to copy/deploy it onto a client machine manually.<<

Man-o-man! I think I'm falling in love ;-)

So when do you think we can Rock & Roll with SilverLight and LINQ? Did you say about six (long) months?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Friday, July 13, 2007 7:14 PM by [ScottGu](#)

Hi Jason,

>>>>> Will they be available for use within my business logic, or will they only be available in the scope of the OnPropertyChanging? Using your example above you could use this.phoneNumber to get the original value before it is changed, but ideally the original value would always be available.

In Beta2 you can use the "GetOriginalEntityState" method from within your DataContext partial class to retrieve an instance of

partial void UpdateCustomer(Customer instance) {

Customer original = this.Customers.GetOriginalEntityState(current);

// todo...

}

You can then do whatever comparison you want with this and the new value to be updated.

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Friday, July 13, 2007 8:00 PM by [Juan MarÃfÂa](#)

Great and extensive post.

You can read in spanish here:

thinkingindotnet.wordpress.com/.../linq-to-sql-4%c2%aa-parte-actualizando-la-base-de-datos

re: LINQ to SQL (Part 4 - Updating our Database)

Saturday, July 14, 2007 7:31 AM by Steve

Thanks for the reply sounds great and beta 2 out so soon you made my weekend. I cant wait for that linq to sql to object to linq to silverlight and back down again demo ;) .

re: LINQ to SQL (Part 4 - Updating our Database)

Saturday, July 14, 2007 7:53 AM by [Majid Shahabfar](#)

Dear Scott,

Thanks for great LINQ to SQL blog series.

Now I want to know which one is faster, using LINQ to SQL or SPROC?

re: LINQ to SQL (Part 4 - Updating our Database)

Saturday, July 14, 2007 9:23 AM by JasonBSteele

Scott,

Thanks for your reply GetOriginalEntityState seems ideal :)

The second part of my previous quote asked "I tried returning one of the Entity objects from a web service but it complained about not being able to serialise circular references. I can't find much ino on the Detach and Attach methods - should I be Detaching it first?"

Also, a few other posts have mentioned using a LINQ to SQL DataContext from a remote (Silverlight) client.

What's the story about about round tripping entity objects. Will we have to wait for Astoria (<http://astoria.mslicelabs.com>) before we have the complete solution?

I appreciate that this subject is worthy of a Part of it's own and maybe outside of what you want to cover.

Many thanks,

Jason

re: LINQ to SQL (Part 4 - Updating our Database)

Sunday, July 15, 2007 12:13 PM by [Zubair.NET!](#)

When extending the DataContext class for using 'Custom Entity Insert/Update/Delete Method Validation', am I limited to using method names as 'InsertOrder' or 'UpdateOrder' or can I change them to something else for instance 'AddOrder' and let the framework trigger this method instead?

re: LINQ to SQL (Part 4 - Updating our Database)

Sunday, July 15, 2007 3:56 PM by Ben Hayat

Hi Scott;

As I'm following your blogs on SQL to LINQ, I can't help it to close or dismiss the other technology that MS is cooking (ADO.Net Entity Framework). I'm a bit perplex as to the existence of these two technologies side by side.

Using the visual designer and creating a datacontext, can give us a well rounded ORM that encapsulates the ORM part and relationships. What does ADO.Net EF offers that SQL To Linq doesn't? What will be the future? Is SQL To LINQ a stepping stone to ADO.Net EF?

Can you shed some light on this issue please?

Thanks!

..Ben

re: LINQ to SQL (Part 4 - Updating our Database)

Sunday, July 15, 2007 6:29 PM by [ScottGu](#)

Hi Jason,

>>>> The second part of my previous quote asked "I tried returning one of the Entity objects from a web service but it complained about not being able to serialise circular references. I can't find much ino on the Detach and Attach methods - should I be Detaching it first?"

I will add this topic to my list to blog about in the future. It is probably worthy of its own blog post (I also need to-do a little more research to get the right answer <g>).

Thanks,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Sunday, July 15, 2007 6:31 PM by [ScottGu](#)

Hi Zubair,

>>>> When extending the DataContext class for using 'Custom Entity Insert/Update/Delete Method Validation', am I limited to using method names as 'InsertOrder' or 'UpdateOrder' or can I change them to something else for instance 'AddOrder' and let the framework trigger this method instead?

The InsertOrder, UpdateOrder, etc methods I talked about above were validation methods - and so can logically be considered internal to the DataContext. They provide you with a single point to add your validation and business rules logic.

You can absolutely add additional methods to the DataContext which are then callable from outside. Just add them to the DataContext's partial class, and you can then implement their logic however you want. When they go to persist an entity, the validation methods will also then fire for you to add domain specific validation logic there if you want.

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Sunday, July 15, 2007 6:40 PM by [ScottGu](#)

Hi Ben,

>>>> Using the visual designer and creating a datacontext, can give us a well rounded ORM that encapsulates the ORM part and relationships. What does ADO.Net EF offers that SQL To Linq doesn't? What will be the future? Is SQL To LINQ a stepping stone to ADO.Net EF?

One of the things about data access is that different database scenarios sometimes require different mapping approaches, and different scenarios often dictate different needs at the ORM layer level.

What we've tried to-do with LINQ is to separate the query language and language integration layer from the data provider layer. This means you can use LINQ to web-services, databases, XML files, or plain old objects using a standard programming model and query syntax. It also means you can plug-in whatever ORM implementation works best for you - whether it is LINQ to SQL, LINQ to Entities, LINQ to NHibernate, etc.

In terms of specific differences between LINQ to SQL and LINQ to Entities, LINQ to Entities provides some richer mapping capabilities (for example: native M:M relationship support and a few other nice things) that you can use. However, for 90%+ of application scenarios I find LINQ to SQL perfect for my needs, and it has great designer and tooling support today that makes it really productive to use.

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Monday, July 16, 2007 6:38 AM by [Eddy Young](#)

Hello Scott,

Thanks for the great series on LINQ. Everything I have read so far is very promising, and I'm looking forward to using LINQ in my applications.

I have a question that I have not seen addressed in any of your blogs: how does LINQ address object equality? Say, you execute two LINQ queries that return the same record, are the two objects representing equal in anyway? Would an update to one affect the other?

Thanks,

Eddy

re: LINQ to SQL (Part 4 - Updating our Database)

Monday, July 16, 2007 9:39 AM by Generating and discovering datacontexts on the fly

I have an application with a static configuration database and an unknown number of content databases with unknown structures. I hold the meta-data for these databases in the configuration database (similar to SharePoint, except the structure of a SharePoint content database is known in advance).

Can I use LINQ to access the content databases? The only thing I can think of that might work is generating sprocs in the content database to present a common interface.

Any suggestions?

re: LINQ to SQL (Part 4 - Updating our Database)

Monday, July 16, 2007 10:04 AM by AzamSharp

Hi Scott,

Thanks for all the answers. I am thinking of using LINQ, DLINQ in one of my projects but I am having a huge layering problem. How do you layer your application when you are using DLINQ? In your examples your are creating the database connection on the client side like the following:

Northwind northwind = new Northwind(connectionString);

I don't want to do this on the client side. But the problem is that even if I put that in the separate project the client will still be able to access it since the classes and the dbml (mapping) files are generated inside the same project. Is there any way to create a DLINQ application in which the client is not able to create a direct connection to the database like Northwind n = new Northwind (connectionString); I prefer the following method:

PersonManager.GetPersonByID(34);

Thanks

LINQ to SQL (Part 5 - Binding UI using the ASP:LinqDataSource Control)

Tuesday, July 17, 2007 2:03 AM by [ScottGu's Blog](#)

Over the last few weeks I've been writing a series of blog posts that cover LINQ to SQL. LINQ to SQL

LINQ to SQL (Part 5 - Binding UI using the ASP:LinqDataSource Control)

Tuesday, July 17, 2007 3:14 AM by [ASP.NET](#)

Over the last few weeks I've been writing a series of blog posts that cover LINQ to SQL. LINQ to SQL

ASP.NET 3.5 will have a LinqDataSource

Tuesday, July 17, 2007 4:43 PM by [Public Sector Developer Weblog](#)

I've been delivering a "What's New for Web Developers in Visual Studio 2008 & the Microsoft ASP.NET

ASP.NET 3.5 will have a LinqDataSource

Tuesday, July 17, 2007 5:25 PM by [Noticias externas](#)

I've been delivering a "What's New for Web Developers in Visual Studio 2008 & the Microsoft

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 18, 2007 2:25 AM by [ScottGu](#)

Hi Eddy,

>>>>>> I have a question that I have not seen addressed in any of your blogs: how does LINQ address object equality? Say, you execute two LINQ queries that return the same record, are the two objects representing equal in anyway? Would an update to one affect the other?

LINQ to SQL provides object equality. This means that if you have two separate LINQ queries that both return the same record, each query will return back the same object instance that represents it. This means that if you update it in one place, the change will be reflected consistently (which is the right behavior).

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Wednesday, July 18, 2007 2:32 AM by [ScottGu](#)

Hi Generating and discovering datacontexts on the fly,

>>>>>> I have an application with a static configuration database and an unknown number of content databases with unknown structures. I hold the meta-data for these databases in the configuration database (similar to SharePoint, except the structure of a SharePoint content database is known in advance).

I don't think LINQ to SQL would work that well for this scenario. But you could build your own LINQ provider that handled this. Bart has a cool post on building a LINQ provider for SharePoint that might help: [community.bartdesmet.net/.../linq-to-sharepoint-announcing-the-0-2-alpha-release.aspx](#)

Hope this helps,
Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 18, 2007 2:34 AM by [ScottGu](#)

Hi AzamSharp,

>>>>>> I don't want to do this on the client side. But the problem is that even if I put that in the separate project the client will still be able to access it since the classes and the dbml (mapping) files are generated inside the same project. Is there any way to create a DLINQ application in which the client is not able to create a direct connection to the database like Northwind n = new Northwind (connectionString); I prefer the following method:

One approach you could use would be to make the NorthwindDataContext internal, and then subclass your own public class that only had a constructor that took no arguments. That way you could prevent a user instantiating the NorthwindDataContext with a custom connectionString.

Hope this helps,
Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 18, 2007 7:02 AM by Andrew Myhre

Hi Scott,

First - thanks for everything you're doing for .Net and people like me.

In a situation where I'm using AJAX over Web Services, let's say my service provides a method SaveCategory(Category myCategory).

The myCategory object is not associated with my DataContext, and if I wanted to use the method equally for Insert and Update operations, there may not be corresponding data in the data store.

Is the DataContext.Category.Attach() method intended for this situation? Does it place the myCategory object within the DataContext, so that I can persist changes?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 18, 2007 8:11 AM by Dave

Thanks for the posts Scott. They have been really helpful. Regarding the DataContext stuff, I will probably just go back to the 'old' way, since there are already a lot of learning curves in this project!

Thanks again!

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 18, 2007 8:12 AM by [Eddy Young](#)

Thank you very much for your answer, Scott. I guess the next question is, how would this work in a distributed system? :-)

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 18, 2007 11:26 AM by SteveP

Hi Scott,

Great post, however there is something i am finding hard to get info about. In ASP.Net is it possible to modify a dataset over a number of postbacks then submit the changes at the end? The problem I see currently is that the same datacontext which was initially used to get the dataset needs to be used when the changes are submitted. Storing the dataset in session as it is being modified does not feel too bad, but would I have to store the datacontext as well? Many thanks.

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 18, 2007 11:30 AM by [ScottGu](#)

Hi SteveP,

>>>>>>> Great post, however there is something i am finding hard to get info about. In ASP.Net is it possible to modify a dataset over a number of postbacks then submit the changes at the end? The problem I see currently is that the same datacontext which was initially used to get the dataset needs to be used when the changes are submitted. Storing the dataset in session as it is being modified does not feel too bad, but would I have to store the datacontext as well? Many thanks.

LINQ to SQL has the ability to disconnect and then later attach a data model object to a DataContext. So what you'd end up doing is rendering the page initially, and then have the DataContext go out of scope. On a subsequent postback you could then attach the DataModel back to a new DataContext for saving it. I'll cover this in a future blog post.

Hope this helps,
Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 18, 2007 11:32 AM by [ScottGu](#)

Hi Andrew,

>>>>>>> In a situation where I'm using AJAX over Web Services, let's say my service provides a method SaveCategory(Category myCategory). The myCategory object is not associated with my DataContext, and if I wanted to use the method equally for Insert and Update operations, there may not be corresponding data in the data store. Is the DataContext.Category.Attach() method intended for this situation? Does it place the myCategory object within the DataContext, so that I can persist changes?

Yep - the Attach() method on the DataContext will be what you are after. I'm going to try and do a dedicated blog post on this in the future to walkthrough how you can use it.

Thanks,
Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, July 18, 2007 12:16 PM by SteveP

thanks for the response. I assume this will be a beta2 feature? as the change tracking is currently being done in the datacontext, i guess there is no way at the moment that the new data context would know what has changed in the data model in the meantime (especially regarding deletes).

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Thursday, July 19, 2007 7:12 PM by James

Hi Scott,

Thanks for this clarification - Is one able to access a query as an object? E.g. can it be passed as a parameter to something else (as in NHibernate)?

The ability to do that is really useful for this scenario: One has a general query, which is reused by methods that just add on their specific criteria before retrieval.

E.g (this it the NH approach, so lacks the sugar of LINQ, but the idea should be clear)

```
IQuery q = GetGeneralQuery();
```

```
q.Add(cond);
```

```
ExecuteQuery(q);
```

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Friday, July 20, 2007 11:24 AM by AzamSharp

>>>One approach you could use would be to make the NorthwindDataContext internal, and then subclass your own public class >>>that only had a constructor that took no arguments. That way you could prevent a user instantiating the >>>NorthwindDataContext with a custom connectionstring.

Hi Scott,

You cannot make the NorthwindDataContext internal and then make a custom class which inherits from the NorthwindDataContext. If you make the NorthwindDataContext class as internal then you cannot have a subclass which is public since it will be more accessible then its base class.

Also, you cannot do any changes in the NorthwindDataContext.designer.cs class since the next time you build the application your changes will be gone.

If seems that DLINQ is forcing a developer to write the data access code in the UI!!

[# LINQ to SQL \(パート 4 - データベースの更新\)](#)

Sunday, July 22, 2007 1:18 AM by [Chica's Blog](#)

LINQ to SQL (パート 4 - データベースの更新)

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Tuesday, July 24, 2007 2:16 PM by [ScottGu](#)

Hi James,

>>>>> Thanks for this clarification - Is one able to access a query as an object? E.g. can it be passed as a parameter to something else (as in NHibernate)? The ability to do that is really useful for this scenario: One has a general query, which is reused by methods that just add on their specific criteria before retrieval. E.g (this it the NH approach, so lacks the sugar of LINQ, but the idea should be clear)

```
IQuery q = GetGeneralQuery();
```

```
q.Add(cond);
```

```
ExecuteQuery(q);
```

Yes - you can pass queries as parameters. They are strongly typed Func<T> objects - so you also preserve type-checking when you do this.

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Thursday, July 26, 2007 4:38 PM by Chien-Hao (Benjamin) Chi

This is awesome! It covers many real world scenarios.

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Tuesday, July 31, 2007 12:57 PM by Alexander

Hi Scott!

I have a question: what result type will Linq to Sql generate for such stored procedure:

```
CREATE PROCEDURE dbo.pr_some
```

```
AS
```

```
SELECT *
```

```
FROM dbo.table1
```

```
SELECT *
```

```
FROM dbo.table2
```

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, August 01, 2007 2:02 AM by [ScottGu](#)

Hi Alexander,

>>>>>> I have a question: what result type will Linq to Sql generate for such stored procedure:

```
CREATE PROCEDURE dbo.pr_some
```

```
AS
```

```
SELECT *
```

```
FROM dbo.table1
```

```
SELECT *
```

```
FROM dbo.table2
```

By default when you use the LINQ to SQL designer the above stored procedure will generate a new entity class whose structure maps to the return type of the SPROC. If you have an existing entity that matches this signature, you can optionally drag the sproc onto it in the designer in order to generate a method that returns a datatype typed as that entity class.

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, August 01, 2007 4:02 AM by Alexander

Hi Scott!

Thank you for your answer, but how can I create entity class that will store data from two select statements? In SqlDataReader I could use NextResult method. Is there anything like this in LINQ to SQL?

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, August 02, 2007 5:45 PM by [erkan saldir](#)

Thank you for great articles

re: LINQ to SQL (Part 4 - Updating our Database)

Friday, August 03, 2007 12:29 PM by Houman

Hello Scott,

How would you retrieve the Category Entity for a Product with ProductID == 2?

Obviously you could search for a product with that ProductID and retrieve the CategoryID then in a second call you could search for the Category with that CategoryID. But these are two calls. Is there a simpler way in one line? Or just in general how would you write this in your style?

Many Thanks,

Houman

re: LINQ to SQL (Part 4 - Updating our Database)

Sunday, August 05, 2007 1:14 PM by bill burrows

Hi Scott,

Great series! Thanks bunch. I am trying the "adding a new order" code from your "Updates across Relationships" section. Whne I try this as a web application, the update works fine. However, if I create a windows application and try the same code, the database does not get updated. Do you have any ideas of why?

Thanks.

re: LINQ to SQL (Part 4 - Updating our Database)

Monday, August 06, 2007 1:31 AM by [ScottGu](#)

Hi Bill,

>>>>> Great series! Thanks bunch. I am trying the "adding a new order" code from your "Updates across Relationships" section. Whne I try this as a web application, the update works fine. However, if I create a windows application and try the same code, the database does not get updated. Do you have any ideas of why?

That sounds pretty odd. Are you sure the code is the same? LINQ to SQL is definitely supported with windows applications as well, so should work fine.

Thanks,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Monday, August 06, 2007 1:35 AM by [ScottGu](#)

Hi Houman,

>>>>>> How would you retrieve the Category Entity for a Product with ProductID == 2?

You should be able to write this code:

```
Category c = northwind.Products.Single(p=>p.ProductID == 2).Select(p=>p.Category);
```

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, August 09, 2007 4:25 AM by [Kangnoz](#)

Hi Scott,

You said,We can add a partial class for our "Order" entity and implement the OnValidate() partial method that will be invoked prior to the entity's values being persisted into the database.

I can't find the invoking code of OnValidate method,after trace the application,I think it invoked at the beginning of SubmitChanges() method. But OnValidate is a private method, how did it invoked?

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, August 09, 2007 1:52 PM by [ScottGu](#)

Hi Kangnoz,

>>>>>> You said,We can add a partial class for our "Order" entity and implement the OnValidate() partial method that will be invoked prior to the entity's values being persisted into the database. I can't find the invoking code of OnValidate method,after trace the application,I think it invoked at the beginning of SubmitChanges() method. But OnValidate is a private method, how did it invoked?

OnValidate is a partial method, and is called from the LINQ to SQL framework automatically. If you look in the code-generated by the LINQ to SQL designer I think you'll see the code where this happens.

Thanks,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Friday, August 10, 2007 4:00 AM by manu

Hi Scott,

I'm trying to change the Default Methods (delete/insert/update) of entity classes, but the fields are disabled!

I'm using Visual Web Developer Express, is it a feauture of Visual Studio?

manu

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, August 16, 2007 11:46 AM by Christian Nielsen

Hi Scott.

I've been looking a bit into designing a layered webapplication using LINQ to SQL in the middle tier. I can't seem to find a good way to handle something as simple as a standard update scenario in a stateless invironment.

Basically, how is a method like this supposed to be implemented:

```
public void Update(Person p)

{

    DataContext db = new DataContext();

    //What goes here?

}
```

re: LINQ to SQL (Part 4 - Updating our Database)

Thursday, August 16, 2007 5:58 PM by [ScottGu](#)

Hi Christian,

>>>>>>> I've been looking a bit into designing a layered webapplication using LINQ to SQL in the middle tier. I can't seem to find a good way to handle something as simple as a standard update scenario in a stateless invironment.

You can use the "Attach()" method on tables to add disconnected entities into the DataContext for tracking. There has been some confusion on how to use this - I'm planning on doing a detailed blog post that describes it in more detail.

Thanks,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Friday, August 24, 2007 5:40 PM by [Dave Sanders](#)

Hi Scott, sorry so many questions today, but I'm finally able to spend a day digging into this stuff.

First question: is there a better place to post questions? :) Is there a newsgroup / list that you and others from MS are actually responding on? I don't typically have good luck on microsoft.public.newsgroups.. :(

Second: how do you get an object to refresh its related objects with what's currently in the database? For example, I have Object A, which is related to Object B via a relationship. I go in and delete object B from the database, but when I ask A for its this.B.Count(), its still coming back as "1". But I can query the DB and see that object B is definitely gone.

thx

Dave (again :)

re: LINQ to SQL (Part 4 - Updating our Database)

Monday, August 27, 2007 8:16 AM by vik

i have bound a GridView with linqDataSource and there is a date column as well. now, i have created a TemplateColumn for displaying and the input of date since i am using AJAXCalendarExtender for date input. All the values in gridview are getting updated except the date. LinqDataSource is set to update automatically. any idea, what i am missing ?

re: LINQ to SQL (Part 4 - Updating our Database)

Monday, August 27, 2007 5:07 PM by [ScottGu](#)

Hi Dave,

>>>>> First question: is there a better place to post questions? :) Is there a newsgroup / list that you and others from MS are actually responding on? I don't typically have good luck on microsoft.public.newsgroups.. :(

The LINQ forum is probably the best place to ask LINQ related questions: forums.microsoft.com/.../ShowForum.aspx

>>>>>> Second: how do you get an object to refresh its related objects with what's currently in the database? For example, I have Object A, which is related to Object B via a relationship. I go in and delete object B from the database, but when I ask A for its this.B.Count(), its still coming back as "1". But I can query the DB and see that object B is definitely gone.

You should be able to call the DataContext.Refresh() method to refresh the current state of the database in your object graphs I believe.

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Monday, August 27, 2007 8:00 PM by Eric

Hi,

What are the different way to update the generated entity classes when the database schema changes?

thanks,

re: LINQ to SQL (Part 4 - Updating our Database)

Tuesday, August 28, 2007 12:53 AM by [JNatUK](#)

Hi,

Thanks for the info very helpful. I created a new LINQ to SQL file using Orcas & when I build my project I get a 'Invalid token 'void' in class, struct, or interface member declaration' on the designer file at the following lines

```
partial void OnLoaded();

    partial void OnValidate();

    partial void OnCreated();
```

This code is generated, how do i get rid of the error?

ta

re: LINQ to SQL (Part 4 - Updating our Database)

Tuesday, August 28, 2007 4:54 PM by [ScottGu](#)

Hi JNatUK,

>>>>>>> Thanks for the info very helpful. I created a new LINQ to SQL file using Orcas & when I build my project I get a 'Invalid token 'void' in class, struct, or interface member declaration' on the designer file at the following lines

Hmmm - are you doing this with Beta2?

Thanks,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Tuesday, August 28, 2007 4:55 PM by [ScottGu](#)

Hi Eric,

>>>>>> What are the different way to update the generated entity classes when the database schema changes?

You can delete and recreate the entity using the LINQ to SQL designer, or chain the SqlMetal utility to run as part of your build process (to recreate the entities each time).

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, August 29, 2007 11:35 AM by Stas

what would be a VB.NET equivalent of this line?

```
Category beverages = northwind.Categories.Single(c => c.CategoryName == 'Beverages');
```

thanks in advance.

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Wednesday, August 29, 2007 12:34 PM by Michael

Hi Scott,

Great stuff - Very flexible and powerful ORM system, and very easy to use.

Two questions:

1 - Is there a bug in the way it deals with default dates in the database by default or am I doing something wrong?

I have a [CreatedDate] column with default GetUTCDate() value in the database. When I generate a LINQ to SQL class from the table and try to do an insert without explicitly setting the CreatedDate property, it tries to put 01/01/0001 into the column instead of letting SQL insert the current UTC date. The insert fails because of the invalid date.

2 - Is there as way to see what SQL that LINQ is generating and have it available to write to the debug window or include in an exception if there is a problem? In the above example, I got the usual invalid date time exception but it was initially difficult to tell what happened because I could not see the SQL or the parameters being passed. I did read that you typically use profiler to do this, but it would be helpful to have this available within the IDE while debugging as we don't have SA access in SQL server 2000 and can't run profiler, and also it would help in a production environment to see the SQL if an exception was thrown.

Thanks,

Michael

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Saturday, September 01, 2007 8:02 PM by JPG

Hi Scott,

In a scenario with a Formview and LinqDataSource,

Is there a way to intercept the insert to modified the value of a specific field base on a Session variable?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Monday, September 03, 2007 3:46 PM by Marcel

HOw can your return the last IDENTITY with LINQ (like SELECT SCOPE_IDENTITY())?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Tuesday, September 04, 2007 11:47 PM by Colin

Hi Scott,

Thanks for all the great content on your blog. I've got a question that I've been unable to find the answer for: what is the story for cross-database CRUD operations in LINQ?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Friday, September 07, 2007 9:49 AM by [JoeBuddha](#)

I'm using Beta 2, and I can't seem to get the database to update. No matter how I set the ObjectTrackingEnabled, the associated databases are still flagged as read-only. I finally had to use the ExecuteCommand method to update by hand. Could I be missing something here?

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Saturday, September 08, 2007 2:53 PM by [ScottGu](#)

Hi Stas,

>>>>>> what would be a VB.NET equivalent of this line? Category beverages = northwind.Categories.Single(c => c.CategoryName == 'Beverages');

Here is the VB equivalent:

```
Dim beverages = northwind.Categories.Single(Function(c) c.CategoryName = "Beverages")
```

Hope this helps,

Scott

[# re: LINQ to SQL \(Part 4 - Updating our Database\)](#)

Saturday, September 08, 2007 2:59 PM by [ScottGu](#)

Hi Michael,

>>>>>> 1 - Is there a bug in the way it deals with default dates in the database by default or am I doing something wrong?I have a [CreatedDate] column with default GetUTCDate() value in the database. When I generate a LINQ to SQL class from the table and try to do an insert without explicitly setting the CreatedDate property, it tries to put 01/01/0001 into the column instead of letting SQL insert the current UTC date. The insert fails because of the invalid date.

That is a good question. Can you send me an email (scottgu@microsoft.com) with details on this? I can then loop you in with folks on the LINQ to SQL team to investigate.

>>>>>>>> 2 - Is there as way to see what SQL that LINQ is generating and have it available to write to the debug window or include in an exception if there is a problem? In the above example, I got the usual invalid date time exception but it was initially difficult to tell what happened because I could not see the SQL or the parameters being passed. I did read that you typically use profiler to do this, but it would be helpful to have this available within the IDE while debugging as we don't have SA access in SQL server 2000 and can't run profiler, and also it would help in a production environment to see the SQL if an exception was thrown.

There is a "Log" property on DataContext objects that you can use to retrieve the raw SQL query and command text being set to the database. I need to think of a way to easily hook this up in the context of an ASP.NET web application - but you should be able to use this. Typically what I personally do is just hook up the SQL Profiler at the database layer to watch what is going on.

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Saturday, September 08, 2007 3:01 PM by [ScottGu](#)

Hi JPG,

>>>>> In a scenario with a Formview and LinqDataSource, Is there a way to intercept the insert to modified the value of a specific field base on a Session variable?

Yes - you can handle an Inserting event on the LinqDataSource or FormView control to inject additional values into the object (pulled from anywhere) before the insert happens.

Hope this helps,

Scott

re: LINQ to SQL (Part 4 - Updating our Database)

Saturday, September 08, 2007 3:31 PM by [ScottGu](#)

Hi Joe,

>>>>>>> I'm using Beta 2, and I can't seem to get the database to update. No matter how I set the ObjectTrackingEnabled, the associated databases are still flagged as read-only. I finally had to use the ExecuteCommand method to update by hand. Could I be missing something here?

Hmm - that is odd. Can you check to make sure your database has a primary key set? That is one possible issue that I can think of that might cause the entities to be read-only.

Thanks,

Scott

Nathan Li's Blog » Blog Archive » Insert/Delete using LINQ to SQL in Visual Studio 2008

Thursday, January 17, 2008 5:27 AM by [Nathan Li's Blog » Blog Archive » Insert/Delete using LINQ to SQL in Visual Studio 2008](#)

Pingback from Nathan Li's Blog » Blog Archive » Insert/Delete using LINQ to SQL in Visual Studio 2008

LINQ TO SQL CanBeNull being ignored

Friday, February 22, 2008 12:58 AM by [Rocks Thoughts](#)

LINQ TO SQL CanBeNull being ignored

Detailed Look: Key components in LINQ to SQL and their Key Roles

Thursday, February 28, 2008 9:19 AM by [Corey Gaudin](#)

As part of this blog, I plan to have an on-going set of articles that takes a detailed look into some

.net tricky. » Blog Archiv » LINQ

Friday, April 04, 2008 11:57 AM by [.net tricky. » Blog Archiv » LINQ](#)

Pingback from .net tricky. » Blog Archiv » LINQ

?????????????????? » Blog Archive » LINQ to SQL??????Part 4 - Updating our Database

Sunday, July 06, 2008 4:57 AM by [????????????????? » Blog Archive » LINQ to SQL??????Part 4 - Updating our Database](#)

Pingback from ?????????????????? » Blog Archive » LINQ to SQL??????Part 4 - Updating our Database

Question on .SubmitChanges

Tuesday, September 30, 2008 10:38 AM by [LINQ in Action roller](#)

Hi everyone, When you call .SubmitChanges, does it perform one UPDATE to the database, or are multiple

Validation rules with LINQ-why should they exist under the App_Code folder?

Thursday, October 02, 2008 3:26 PM by [LINQ in Action roller](#)

Hi everyone, I was reading Scott's blog on validation rules [weblogs.asp.net/.../linq-to-sql-part-4-updating-our-database.aspx](#)

Question on LINQ - Why use validation rule on entire change list for transaction?

Friday, October 03, 2008 3:02 PM by [LINQ in Action roller](#)

Looking at the following blog (validation rules), [weblogs.asp.net/.../linq-to-sql-part-4-updating-our-database.aspx](#)

New and Notable 177

Tuesday, December 02, 2008 8:38 PM by [Sam Gentile's Blog](#)

Multithreading and Concurrency Software Transactional Memory Part IV - Thread-Bound Transactions Software Transactional Memory Part V - Integration with System.Transactions Parallel LINQ Restating the Concurrency Problem Herb Sutter is starting a new

转载:LINQ to SQL更新数据库操作

Tuesday, December 09, 2008 7:39 AM by [姜敏](#)

LINQ to SQL更新数据库操作

<#> **ASP.NET MVC Archived Buzz, Page 1**

Monday, January 05, 2009 10:38 AM by [ASP.NET MVC Archived Buzz, Page 1](#)

Pingback from [ASP.NET MVC Archived Buzz, Page 1](#)

[Terms of Use](#)