What You Can Do With LINQ to SQL

Article • 09/15/2021 • 3 minutes to read

LINQ to SQL supports all the key capabilities you would expect as a SQL developer. You can query for information, and insert, update, and delete information from tables.

Selecting

Selecting (*projection*) is achieved by just writing a LINQ query in your own programming language, and then executing that query to retrieve the results. LINQ to SQL itself translates all the necessary operations into the necessary SQL operations that you are familiar with. For more information, see LINQ to SQL.

In the following example, the company names of customers from London are retrieved and displayed in the console window.

C#

```
// Northwnd inherits from System.Data.Linq.DataContext.
Northwnd nw = new Northwnd(@"northwnd.mdf");
// or, if you are not using SQL Server Express
// Northwnd nw = new
Northwnd("Database=Northwind;Server=server_name;Integrated
Security=SSPI");

var companyNameQuery =
    from cust in nw.Customers
    where cust.City == "London"
    select cust.CompanyName;

foreach (var customer in companyNameQuery)
{
    Console.WriteLine(customer);
}
```

Inserting

To execute a SQL `Insert`, just add objects to the object model you have created, and call SubmitChanges on the DataContext.

In the following example, a new customer and information about the customer is added to the `Customers` table by using InsertOnSubmit.

C#

```
// Northwnd inherits from System.Data.Linq.DataContext.
Northwnd nw = new Northwnd(@"northwnd.mdf");

Customer cust = new Customer();
cust.CompanyName = "SomeCompany";
cust.City = "London";
cust.CustomerID = "98128";
cust.PostalCode = "55555";
cust.Phone = "555-555-5555";
nw.Customers.InsertOnSubmit(cust);

// At this point, the new Customer object is added in the object model.
// In LINQ to SQL, the change is not sent to the database until
// SubmitChanges is called.
nw.SubmitChanges();
```

Updating

To `Update` a database entry, first retrieve the item and edit it directly in the object model. After you have modified the object, call SubmitChanges on the DataContext to update the database.

In the following example, all customers who are from London are retrieved. Then the name of the city is changed from "London" to "London - Metro". Finally, SubmitChanges is called to send the changes to the database.

C#

```
Northwnd nw = new Northwnd(@"northwnd.mdf");

var cityNameQuery =
    from cust in nw.Customers
    where cust.City.Contains("London")
    select cust;

foreach (var customer in cityNameQuery)
{
    if (customer.City == "London")
    {
        customer.City = "London - Metro";
    }
}
nw.SubmitChanges();
```

Deleting

To `Delete` an item, remove the item from the collection to which it belongs, and then call

SubmitChanges on the DataContext to commit the change.

> **Note**
>
> LINQ to SQL does not recognize cascade-delete operations. If you want to delete a row in a table that has constraints against it, see **How to: Delete Rows From the Database**.

In the following example, the customer who has `CustomerID` of `98128` is retrieved from the database. Then, after confirming that the customer row was retrieved, DeleteOnSubmit is called to remove that object from the collection. Finally, SubmitChanges is called to forward the deletion to the database.

C#

```
Northwnd nw = new Northwnd(@"northwnd.mdf");
var deleteIndivCust =
    from cust in nw.Customers
    where cust.CustomerID == "98128"
    select cust;

if (deleteIndivCust.Count() > 0)
{
    nw.Customers.DeleteOnSubmit(deleteIndivCust.First());
    nw.SubmitChanges();
}
```

See also

- Programming Guide
- The LINQ to SQL Object Model
- Getting Started