تقنية LINQ

القسم النظري من تقنية LINQ

LINQ :

هي تقنية برمجية أنتجتها شركة مايكروسوفت وأصدرتها في 2007 ضمن إطار العمل netframework 3.5

الغرض منها : تسهيل عملية معالجة البيانات مهما كان مصدرها

أي أننا نستطيع عن طريق هذه التقنية إجراء عمليات الاستعلام و الإضافة و الحذف و التعديل و الترتيب و التجميع و الكثير من العمليات التي تمكننا إياها لغة SQL على قواعد البيانات الخاصة بها و بأسلوب سهل و بسيط

ونشير إلى موضوع قد يخطر في الذهن :

كيف نستخدم هذه التقنية ؟

بكل بساطة يمكننا استخدام لغة C# او لغة VB لإجراء عمليات Linq
لذلك يمكنك أن تتعلم تقنية linq مباشرةً دون أن يكون لك خلفية في تعليمات sql

ميزات تقنية LINQ :

1- توحيد مصادر البيانات و التعامل معها بآلية عمل واحدة

أي أننا نستطيع إجراء استعلام من sql و xml و object و الكثير من أنواع مصادر البيانات بنفس الطريقة

2- إعطاء قوة كبيرة للغة XML :

عن طريق هذه التقنية فقد جعلنا لغة XML لغة يمكن أن نقارنها بـ SQL

هل تعلم أن لغة XMLعن طريق تقنياتها الحديثة أصبحت تحتل مركز كبير في الاستخدام في أجهزة الأندرويد و ألعاب الكومبيوتر و الكثير من البرامج و تحديثاتها

ميزة يجب ذكرها عن XML تخص تقنيتنا التي نتحدث عنها :

وأنت تتعلم لغة SQL سواء SQL server أو Oracle سوف تتفاجئ بأخر مطافك أنك تتعلم لغات عالية من كل النواحي ولكن لها سلبيات:

أهم سلبية أنك لا يمكنك استخدام مثل هذه القواعد في المشاريع الصغيرة مثل إدارة متجر أو صيدلية لسبب صغير وهو اقتلاع الداتا بيز بعد الانتهاء

أي أنك بحاجة لتنصيب نفس مدير قاعدة البيانات مثل Sql serverعند الزبون و بحاجة ان تقتلع الداتا بيز المنشئة و زرعها عند جهاز الزبون

هذه ليست مشكلة بقدر مشكلة أنك لست متوقع ماهو الكم الهائل من الأخطاء الذي قد يحدث عند جهاز الزبون لأنك ببساطة كنت قد برمجت الداتا بيز لديك ولم تبرمجها عند الزبون

فلربما الزبون يمتلك جهاز ضعيف لا يتحمل مدير قاعدة البيانات ولربما أيضاً تحدث أخطاء بسبب مشاكل لدى جهاز الزبون فتضطر إلى إصلاح مشاكل جهازه قبل إعطاء الداتا بيز
ولربما تحدث مشاكل في تنصيب مدير قاعدة البيانات و لربما و لربما

ووصلت أخيراً مدير قواعد البيانات إلى حل المشكلة في وضع مدير قاعدة البيانات على سيرفر بعيد و الكثير من الأمور المعقدة الأخرى ... لن ندخل في تفاصيلها
وهنا تكمن لغة XML التي تتميز بتقنية PLUG & PLAY أي لا تحتاج لتنصيب أي شيء ولا تحتاج لاقتلاع أي قاعدة لا تحتاج سوى إعطاء البرنامج الذي قمت ببرمجته مع ملفاته و تنقلها للزبون و تقوم بتشغيل البرنامج بشكل مباشر و تتعامل مع قواعد البيانات من دون أي مشاكل

ما علاقة تقنيتنا بالموضوع ؟

يجدر على المنصت للموضوع أن يدري أن لغة XML هي لغة صامتة أي تحتاج لمحرك لها وتقنية LINQ هي المحرك لها أي هي التي ستجعلها مثل باقي لغات قاعدة البيانات الكبيرة حيث أنها هي لغة الاستعلام و الإضافة و الحذف ووو...

القسم العملي من تقنية LINQ

سوف يتم عرض في السلايدات القادمة كيفية التعامل مع تقنية LINQ عبر لغة C#
وسيتم استخدام اللوائح و المصفوفات و الـ Objects في شرح تقنيتنا

يجدر بنا الذكر أن تقنية LINQ لها اشتقاقات مثل LINQ TO XML وهي تقنية LINQ
المطبقة على ملفات الـ XML

في هذا الملف لن يتم شرح LINQ TO XML
و الجدير بالذكر أيضاً سوف نركز على إجراء الاستعلامات المشهورة في sql فقط

هناك شكلان للتعامل مع تقنية LINQ:

1- Query-Syntax:

وهي طريقة لإجراء الاستعلامات من مصدر البيانات و هي شبيهة جداً بلغة استعلام SQL

2- Methods-Syntax:

وهي طريقة لإجراء الاستعلامات من مصدر البيانات عن طريق الوظائف الممتدة

الكثير من مستخدمي تقنية LINQ يجدون الشكل الثاني أسهل من الشكل الأول و خصوصاً الذين لم يتعاملوا مع SQL
أما من كان قد تعامل مع SQL فسيجد الشكل الأول أسهل من الشكل الثاني

```
static void Main(string[] args)
{
    List<string> names = new List<string>() { "Ahmad", "Zaher", "Hassan", "Husam", "Yasser" };
    //query syx:
    var query1 = from x in names
                 select x;

    Console.WriteLine("query syx:");
    foreach (string x in query1)
    {
        Console.Write(x + "  ");
    }
    Console.WriteLine();
    Console.WriteLine("-----------------------------------");
    //Method syx:
    var query2 = names;

    Console.WriteLine("Methods syx:");
    foreach (string x in query2)
    {
        Console.Write(x + "  ");
    }

    Console.WriteLine();
    Console.ReadKey();
}
```

استعلام بسيط دون شرط

قمنا بتعريف لائحة لنقوم بإجراء استعلام عليها دون أي شروط

```
List<string> names = new List<string>()
{ "Ahmad", "Zaher", "Hassan", "Husam", "Yasser","Suliman" };
```

```
//query syx:
var query1 = from x in names
             select x;
```

```
//Method syx:
var query2 = names;
```

```
query syx:
Ahmad  Zaher  Hassan  Husam  Yasser
-----------------------------------
Methods syx:
Ahmad  Zaher  Hassan  Husam  Yasser
```

Where

كيف نجري استعلام مع شرط معين :

الاستعلام التالي يقوم بجلب الأسماء التي يزيد طولها عن 5 أحرف و الخاصية Length لقياس عدد الأحرف

```
List<string> names = new List<string>()
{ "Ahmad", "Zaher", "Hassan", "Husam", "Yasser","Suliman" };
```

```
//query syx:
var query1 = from x in names
             where x.Length>5
            select x;
```

```
//Method syx:
var query2 = names.Where(x =>x.Length>5);
```

```
query syx:
Hassan  Yasser  Suliman
--------------------------------
Methods syx:
Hassan  Yasser  Suliman
```

StartWith()

الاستعلام التالي يقوم بجلب الأسماء التي تبدأ بحرف H:

الدالة StartWith() : لجلب الكلمات التي تبدأ بحرف أو أحرف معينة

```
List<string> names = new List<string>()
{ "Ahmad", "Zaher", "Hassan", "Husam", "Yasser","Suliman" };

    //query syx:
    var query1 = from x in names
                 where x.StartsWith("H")
                select x;

//Method syx:
var query2 = names.Where(x =>x.StartsWith("H"));
```

```
query syx:
Hassan  Husam
--------------------------------
Methods syx:
Hassan  Husam
```

Contains()

الاستعلام التالي يقوم بجلب الأسماء التي تحتوي على حرفين SS:

الدالة Contains() : لجلب الكلمات التي تبدأ على حرف أو أحرف معينة

```
List<string> names = new List<string>()
{ "Ahmad", "Zaher", "Hassan", "Husam", "Yasser","Suliman" };

 //query syx:
 var query1 = from x in names
              where x.Contains("ss")
             select x;

//Method syx:
var query2 = names.Where(x =>x.Contains("ss"));
```

```
query syx:
Hassan  Yasser
--------------------------------
Methods syx:
Hassan  Yasser
```

Take()

الاستعلام التالي يقوم بجلب أول اسمين يحويان على حرف m :

الدالة take(n) : لجلب أول n عنصر

```
List<string> names = new List<string>()
{ "Ahmad", "Zaher", "Hassan", "Husam", "Yasser","Suliman" };
```

```
//Method syx:
var query2 = names.Where(x =>x.Contains("m")).Take(2);
```

```
Methods syx:
Ahmad  Husam
```

AND - OR

يمكن أن تحتوي الشروط على AND و OR

```
//query syx:
var query1 = from x in names
             where x.Contains("m") && x.Length>5
            select x;
```

```
//Method syx:
var query2 = names.Where(x =>x.Contains("m")&& x.Length > 5);
```

Objects

لندخل الآن في الـ Objects:

قمنا بتعريف كلاس employee و اشتقنيا Object منه في قسم الـ Main لإجراء بعض الاستعلامات عليه

```
class employee
{
    7 references
    public int id { get; set; }
    6 references
    public string name { get; set; }
    6 references
    public string lang { get; set; }
}
```

```
employee[] emp ={
                    new employee(){id = 1, name = "Ahmad", lang="C#"},
                    new employee(){id = 2, name = "Zaher", lang="Java"},
                    new employee(){id = 3, name = "Hassan", lang="Python"},
                    new employee(){id = 4, name = "Husam", lang="C#"},
                    new employee(){id = 5, name = "Yasser", lang="Java"},
                    new employee(){id = 6, name = "Suliman", lang="C++"}
                };
```

Objects

الاستعلام التالي يقوم بجلب جميع البيانات من الـ collection السابقة دون شروط

```
//query syx:
var query1 = from x in emp
             select x;

foreach (var x in query1)
{
    Console.WriteLine(x.id + "  "+x.name+"  "+x.lang);
}

//Method syx:
var query2 = emp;

foreach (var x in query2)
{
    Console.WriteLine(x.id + "  " + x.name + "  " + x.lang);
}
```

```
query syx:
1  Ahmad  C#
2  Zaher  Java
3  Hassan  Python
4  Husam  C#
5  Yasser  Java
6  Suliman  C++

--------------------------------
Methods syx:
1  Ahmad  C#
2  Zaher  Java
3  Hassan  Python
4  Husam  C#
5  Yasser  Java
6  Suliman  C++
```

OrderBy

الاستعلام التالي : يقوم بجلب البيانات مع ترتيبهم حسب الاسم

حيث يعتمد الترتيب حسب الأحرف الإنكليزية من A – Z

حيث تفيد orderby في ذلك

```
employee[] emp ={
                    new employee(){id = 1, name = "Ahmad", lang="C#"},
                    new employee(){id = 2, name = "Zaher", lang="Java"},
                    new employee(){id = 3, name = "Hassan", lang="Python"},
                    new employee(){id = 4, name = "Husam", lang="C#"},
                    new employee(){id = 5, name = "Yasser", lang="Java"},
                    new employee(){id = 6, name = "Suliman", lang="C++"}
                };


  //query syx:
  var query1 = from x in emp
               orderby(x.name)
               select x;


  //Method syx:
  var query2 = emp.OrderBy(x => x.name);
```

```
query syx:
1  Ahmad  C#
3  Hassan  Python
4  Husam  C#
6  Suliman  C++
5  Yasser  Java
2  Zaher  Java

-------------------------------

Methods syx:
1  Ahmad  C#
3  Hassan  Python
4  Husam  C#
6  Suliman  C++
5  Yasser  Java
2  Zaher  Java
```

descending

الاستعلام التالي : يقوم بجلب البيانات مع ترتيبهم ترتيب عكسي حسب الاسم

حيث يعتمد الترتيب حسب الأحرف الإنكليزية من A – Z

```
employee[] emp ={
                    new employee(){id = 1, name = "Ahmad", lang="C#"},
                    new employee(){id = 2, name = "Zaher", lang="Java"},
                    new employee(){id = 3, name = "Hassan", lang="Python"},
                    new employee(){id = 4, name = "Husam", lang="C#"},
                    new employee(){id = 5, name = "Yasser", lang="Java"},
                    new employee(){id = 6, name = "Suliman", lang="C++"}
                };

    //query syx:
    var query1 = from x in emp
                 orderby(x.name)descending
                 select x;



    //Method syx:
    var query2 = emp.OrderByDescending(x => x.name);
```

```
query syx:
2  Zaher  Java
5  Yasser  Java
6  Suliman  C++
4  Husam  C#
3  Hassan  Python
1  Ahmad  C#

---------------------------------
Methods syx:
2  Zaher  Java
5  Yasser  Java
6  Suliman  C++
4  Husam  C#
3  Hassan  Python
1  Ahmad  C#
```

ThenBy

الاستعلام التالي : يقوم بجلب البيانات مع ترتيبهم حسب لغة البرمجة ثم النظر إلى الـ id

```
employee[] emp ={
                    new employee(){id = 1, name = "Ahmad", lang="C#"},
                    new employee(){id = 2, name = "Zaher", lang="Java"},
                    new employee(){id = 3, name = "Hassan", lang="Python"},
                    new employee(){id = 4, name = "Husam", lang="C#"},
                    new employee(){id = 5, name = "Yasser", lang="Java"},
                    new employee(){id = 6, name = "Suliman", lang="C++"}
                };
```

```
//query syx:
var query1 = from x in emp
             orderby x.lang,x.id
             select x;

//Method syx:
var query2 = emp.OrderBy(x => x.lang).ThenBy(x => x.id);
```

```
query syx:
1  Ahmad  C#
4  Husam  C#
6  Suliman  C++
2  Zaher  Java
5  Yasser  Java
3  Hassan  Python

---------------------------------

Methods syx:
1  Ahmad  C#
4  Husam  C#
6  Suliman  C++
2  Zaher  Java
5  Yasser  Java
3  Hassan  Python
```

GroupBy

الاستعلام التالي : يقوم بجلب البيانات مع تجميعهم حسب لغة البرمجة

```
//query syx:
var query1 = from x in emp
             group x by x.lang;

Console.WriteLine("query syx:");
foreach (var x in query1)
{
    Console.WriteLine(x.Key);
    Console.WriteLine("-----------------------------------");
    foreach (var item in x)
    {
        Console.WriteLine(item.id + "  " + item.name);
    }
    Console.WriteLine("-----------------------------------");
}
```

```
query syx:
C#
-----------------------------------
1  Ahmad
4  Husam
-----------------------------------
Java
-----------------------------------
2  Zaher
5  Yasser
-----------------------------------
Python
-----------------------------------
3  Hassan
-----------------------------------
C++
-----------------------------------
6  Suliman
-----------------------------------
```

GroupBy

الاستعلام التالي : يقوم بجلب البيانات مع تجميعهم حسب لغة البرمجة

```
//Method syx:
var query2 = emp.GroupBy(x => x.lang);


Console.WriteLine("Methods syx:");
foreach (var x in query2)
{
    Console.WriteLine(x.Key);
    Console.WriteLine("-----------------------------------");
    foreach (var item in x)
    {
        Console.WriteLine(item.id + "  " + item.name);
    }
    Console.WriteLine("-----------------------------------");
}
```

```
-----------------------------------
Methods syx:
C#
-----------------------------------
1  Ahmad
4  Husam
-----------------------------------
Java
-----------------------------------
2  Zaher
5  Yasser
-----------------------------------
Python
-----------------------------------
3  Hassan
-----------------------------------
C++
-----------------------------------
6  Suliman
-----------------------------------
```

Join

قمنا باشتقاق غرضين مختلفين من كلاسين مختلفين وسوف نربط بينهما عن طريق الـ id

```
employee[] emp ={
                    new employee(){id = 1, name = "Ahmad", lang="C#"},
                    new employee(){id = 2, name = "Zaher", lang="Java"},
                    new employee(){id = 3, name = "Hassan", lang="Python"},
                };
salary[] sal    ={
                    new salary(){id = 1, sal = 50000},
                    new salary(){id = 2, sal = 60000},
                    new salary(){id = 4, sal = 70000}
                };
```

Join

الاستعلام التالي يقوم بربط المجموعتين مع بعضهما

```
//query syx:
var query1 = from e in emp
             join s in sal on e.id equals s.id
             select new { employee = e, salary = s };
```

```
Console.WriteLine("query syx:");
foreach (var x in query1)
{
    Console.WriteLine(x.employee.id + " - " + x.employee.name+" : "+x.salary.sal);
}
```

```
query syx:
1 - Ahmad : 50000
2 - Zaher : 60000
```

Join

الاستعلام التالي يقوم بربط المجموعتين مع بعضهما

```
//Method syx:
var query2 = emp.Join(sal, e => e.id, s =>
    s.id, (e, s) => new { employee = e, salary = s });


Console.WriteLine("Methods syx:");
foreach (var x in query2)
{
    Console.WriteLine(x.employee.id + " - " + x.employee.name + " : " + x.salary.sal);
}
```

```
Methods syx:
1 - Ahmad : 50000
2 - Zaher : 60000
```

Concat,Distinct,Union,Intersect,Except

```
string[] group1 = { "Ahmad", "Foad", "Zaher", "Hassan" };
string[] group2 = { "Fares", "Ahmad", "Hassan", "Khaled" };
//Concat & (Concat & Distinct) & Union & Intersect & Except
var Concat = group1.Concat(group2);
var ConcatD = group1.Concat(group2).Distinct();
var Union = group1.Union(group2);
var Intersect = group1.Intersect(group2);
var Except = group1.Except(group2);
```

Concat: لدمج مجموعتين مع التكرار و تستخدم Distinct لحذف المكرر

Union: لدمج مجموعتين مع حذف التكرار

Intersect: لجلب المكرر بين المجموعتين

Except: لجلب الغير مكرر من المجموعة الأولى في الثانية

```
Concat:
Ahmad  Foad  Zaher  Hassan  Fares  Ahmad  Hassan  Khaled
----------------------------
ConcatDistinct:
Ahmad  Foad  Zaher  Hassan  Fares  Khaled
----------------------------
Union:
Ahmad  Foad  Zaher  Hassan  Fares  Khaled
----------------------------
Intersect:
Ahmad  Hassan
----------------------------
Except:
Foad  Zaher
```

Concat,Distinct,Union,Intersect,Except

```
Console.WriteLine("Concat:");
foreach (var x in Concat)
{
    Console.Write(x+"  ");
}
Console.WriteLine("\n----------------------------");

Console.WriteLine("ConcatDistinct:");
foreach (var x in ConcatD)
{
    Console.Write(x + "  ");
}
Console.WriteLine("\n----------------------------");

Console.WriteLine("Union:");
foreach (var x in Union)
{
    Console.Write(x + "  ");
}
Console.WriteLine("\n----------------------------");

Console.WriteLine("Intersect:");
foreach (var x in Intersect)
{
    Console.Write(x + "  ");
}
Console.WriteLine("\n----------------------------");
```

```
Concat:
Ahmad  Foad  Zaher  Hassan  Fares  Ahmad  Hassan  Khaled
----------------------------
ConcatDistinct:
Ahmad  Foad  Zaher  Hassan  Fares  Khaled
----------------------------
Union:
Ahmad  Foad  Zaher  Hassan  Fares  Khaled
----------------------------
Intersect:
Ahmad  Hassan
----------------------------
Except:
Foad  Zaher
```

FirstOrDefault,LastOrDefault

FirstOrDefault : لجلب أول عنصر في المجموعة

LastOrDefault : لجلب أخر عنصر في المجموعة

```
string[] group1 = { "Ahmad", "Foad", "Zaher", "Hassan" };
var first = group1.FirstOrDefault();
var last = group1.LastOrDefault();
Console.WriteLine("first = {0}\nlast = {1}", first, last);
```

وتفيد OrDefault لتجنب الوقوع في خطأ أثناء التنفيذ إن لم يكن هناك عناصر يجلبها
وخاصةً عندما تكون المجموعة فارغة

```
first = Ahmad
last = Hassan
```

Max , Min , Count , Average , Sum

Max : لجلب أكبر قيمة في المجموعة

Min : لجلب أصغر قيمة في المجموعة

Sum : لحساب مجموع عناصر المجموعة

Count : لحساب عدد العناصر في المجموعة

Average : لحساب المتوسط الحسابي لعناصر المجموعة

```
int[] group1 = { 5,8,9,6,8,12 };
var max = group1.Max();
var count = group1.Count();
var min = group1.Min();
var average = group1.Average();
var sum = group1.Sum();

Console.WriteLine("max ={0}\nmin ={1}\ncount ={2}\naverage ={3}\nsum ={4}"
    , max, min, count, average, sum);
```

```
max =12
min =5
count =6
average =8
sum =48
```