عنوان: **نحوه‌ی صحیح فراخوانی SQL Aggregate Functions حین استفاده از LINQ**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۸/۱۳ ۱۸:۳۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: LINQ

SQL Aggregate Functions
که مد نظر شما هستند مانند Sum ، Max ، Min و امثال آن. بحث LINQ هم زمانیکه از الگوی Repository استفاده شود مستقل از نوع ORM مورد نظر خواهد شد؛ بنابراین در اینجا مقصود از LINQ می‌تواند LINQ to NHibernate ، LINQ to Entities ، LINQ to SQL و کلا هر نوع ORM دیگری با پشتیبانی از LINQ باشد.
صورت مساله هم این است: آیا نوشتن عبارت LINQ ایی به شکل زیر صحیح است؟

```
decimal amount = respository.Transactions
                                    .Where(t=>t.TransactionDate>new DateTime(2010,10,13))
                                    .Sum(t=>t.Amount);
```

پاسخ: خیر!
توضیحات:
عبارت LINQ فوق در نهایت به شکل زیر ترجمه خواهد شد:

```
-- Region Parameters
-- @p0: DateTime [2010/10/13 12:00:00 ظ.ق]
-- EndRegion
SELECT SUM([t0].[Amount]) AS [value]
FROM [Transactions] AS [t0]
WHERE [t0].[TransactionDate] > @p0
```

و اتفاقا در این سیستم پس از تاریخ 13/10/2010 هیچ تراکنشی ثبت نشده است؛ بنابراین خروجی این کوئری null خواهد بود و نه صفر. همینجا است که یکی از استثناهای زیر صادر شده و ادامه‌ی برنامه با مشکل مواجه خواهد شد:

```
- System.InvalidOperationException: The cast to value type 'decimal' failed because the materialized
value is null.
- InvalidOperationException: The null value cannot be assigned to a member with type decimal which is a
non-nullable value type.
```

مشکل هم از اینجا ناشی می‌شود که متغییری از نوع deciaml یا int و امثال آن، مقدار دریافتی نال را نمی‌پذیرند. برای رفع این مشکل باید عبارت LINQ فوق به صورت زیر بازنویسی شود (و اهمیتی هم ندارد که Sum است یا Max یا Avg و غیره؛ در مورد بکارگیری تمام SQL Aggregate Functions در یک عبارت LINQ ، این مورد باید لحاظ گردد):

```
decimal amount = respository.Transactions
```

```
.Where(t=>t.TransactionDate>new DateTime(2010,10,13))
```

```
.Sum(t=>
```

```
(?decimal)
```

```
t.Amount)
```

```
0??
```

```
;
```

دقیقا به همین علت است که در دات نت، nullable types تعریف شده‌اند. امکان ذخیره سازی null در یک متغیر برای مثال از نوع decimal وجود ندارد اما نوع decimal? (و یا Nullable<decimal> به بیانی دیگر) این قابلیت را دارد.
شاید بگوئید که در اینجا با تغییر تعریف متغیر به decimal? amount مشکل حل می‌شود، اما خیر. تعریف extension method مربوط به sum به صورت زیر است:

```
public static

TResult

Sum<TSource>(

this IQueryable<TSource> source,

Expression<Func<TSource,

TResult

>> selector)
```

در این تعریف به TResult دقت نمائید؛ هم بیانگر نوع خروجی نهایی متد و هم مشخص سازنده‌ی نوع پارامتری است که خروجی Lambda Expression را تشکیل می‌دهد. به این معنا که سی شارپ، TResult را از lambda expression دریافت کرده و خروجی Sum را بر همان مبنا و نوع تشکیل می‌دهد. بنابراین برای دریافت خروجی nullable باید TResult ایی nullable را همانند مثال فوق ایجاد کنیم.

خلاصه بحث:
اگر در کدهای LINQ خود که با بانک اطلاعاتی سر و کار دارند از معادل‌های SQL Aggregate Functions استفاده کرده‌اید، آن‌ها را یافته و نکته‌ی nullable TResult فوق را به آن‌ها اعمال کنید؛ در غیر اینصورت منتظر باشید تا روزی برنامه شما به سادگی کرش کند.