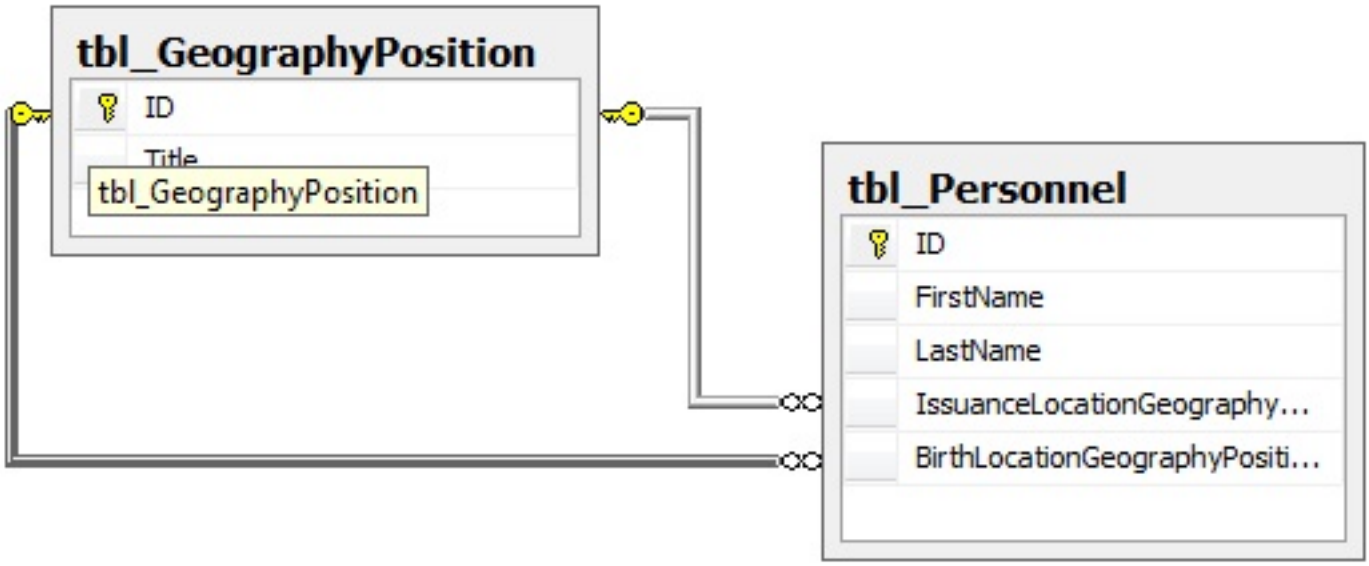
عنوان: **شبیه سازی outer Join در entity framework**
نویسنده: شاهد محمودی
تاریخ: ۱۳۹۲/۰۸/۲۱ ۹:۲۰
آدرس: www.dotnettips.info
برچسب‌ها: Entity framework, join, left outer join, right outer join

فرض کنید دو جدول پرسنل و شهر را در دیتا بیس خود دارید و 2 بار کد شهر در جدول پرسنل ارتباط داده شده که یکی برای محل تولد و دیگری برای محل صدور
و میخواهید توسط outer join لیست تمامی پرسنل و محل تولد و محل صدور را (در صورت وجود) داشته باشید.



در sql گرفتن نتیجه ذکر شده بصورت زیر به راحتی قابل انجام است

```
SELECT
   dbo.tbl_Personnel.ID,
   dbo.tbl_Personnel.FirstName,
   dbo.tbl_Personnel.LastName,
   dbo.tbl_GeographyPosition.Title AS IssuanceLocation,
   tbl_GeographyPosition_1.Title AS BirthLocation
FROM  dbo.tbl_Personnel LEFT OUTER JOIN
          dbo.tbl_GeographyPosition AS tbl_GeographyPosition_1 ON
          dbo.tbl_Personnel.BirthLocationGeographyPositionID = tbl_GeographyPosition_1.ID LEFT OUTER
JOIN
          dbo.tbl_GeographyPosition ON dbo.tbl_Personnel.IssuanceLocationGeographyPositionID =
dbo.tbl_GeographyPosition.ID
```

اما در ef با توجه به خواص راهبری کمتر از join استفاده میکنیم در ضمن به هیچ وجه از Left Outer Join و Right Outer Join استفاده نمی‌شود و باید کوئری فوق را با کد زیر شبیه سازی کرد

```
var contex = new PersonnelEntities();
var Query = from Personnel in contex.tbl_Personnel
                        join IssuanceLocation in contex.tbl_GeographyPosition on
                              Personnel.IssuanceLocationGeographyPositionID equals IssuanceLocation.ID
into AIssuanceLocation
                        from IL in AIssuanceLocation.DefaultIfEmpty()
                        join BirthLocation in contex.tbl_GeographyPosition on
                              Personnel.BirthLocationGeographyPositionID equals BirthLocation.ID into
ABirthLocation
                        from BL in ABirthLocation.DefaultIfEmpty()
                        //where
                        select new
                        {
                            Personnel.ID,
```

```
                            Personnel.FirstName,
                            Personnel.LastName,
                            IssuanceLocation = IL.Title,
                            BirthLocation = BL.Title
                        };
```

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۲۱ ۱۳:۲۱

جهت تکمیل بحث، اگر مدل‌های برنامه به این صورت باشند (محل تولد اجباری است و Id کلید خارجی آن نال پذیر نیست؛ به همراه محل صدور اختیاری، که Id نال پذیر دارد):

```
public class Place
    {
        public int Id { set; get; }
        public string Name { set; get; }

        public virtual ICollection<Person> Personnel { set; get; }
    }

    public class Person
    {
        public int Id { set; get; }
        public string FirstName { set; get; }
        public string LastName { set; get; }

        [ForeignKey("BirthPlaceId")]
        public virtual Place BirthPlace { set; get; }
        public int BirthPlaceId { set; get; }

        [ForeignKey("IssuanceLocationId")]
        public virtual Place IssuanceLocation { set; get; }
        public int? IssuanceLocationId { set; get; }
    }
```

با این Context :

```
public class MyContext : DbContext
    {
        public DbSet<Place> Places { get; set; }
        public DbSet<Person> Personnel { get; set; }

        public MyContext()
        {
            this.Database.Log = sql => Console.WriteLine(sql);
        }
    }
```

آنگاه خروجی کوئری ذیل (که یک include دارد روی خاصیت راهبری که مقدار Id کلید خارجی آن ممکن است نال باشد (محل صدور) و نه مورد دومی که Id غیرنال پذیر دارد (محل تولد))

```
context.Personnel.Include(x => x.IssuanceLocation)
```

معادل خواهد بود با (left outer join به صورت خودکار تشکیل شده)

```
SELECT
    [Extent1].[Id] AS [Id],
    [Extent1].[FirstName] AS [FirstName],
    [Extent1].[LastName] AS [LastName],
    [Extent1].[BirthPlaceId] AS [BirthPlaceId],
    [Extent1].[IssuanceLocationId] AS [IssuanceLocationId],
    [Extent2].[Id] AS [Id1],
    [Extent2].[Name] AS [Name],
    [Extent1].[Place_Id] AS [Place_Id]
    FROM  [dbo].[People] AS [Extent1]
    LEFT OUTER JOIN [dbo].[Places] AS [Extent2] ON [Extent1].[IssuanceLocationId] = [Extent2].[Id]
```

و خروجی کوئری زیر که DefaultIfEmpty را هم لحاظ کرده و join نویسی صریحی هم دارد (مطابق مقاله فوق):

```
var query = from personnel in context.Personnel
                            join issuanceLocation in context.Places on
                                  personnel.IssuanceLocationId equals issuanceLocation.Id into
aIssuanceLocation
                            from IL in aIssuanceLocation.DefaultIfEmpty()
                            join birthLocation in context.Places on
                                  personnel.BirthPlaceId equals birthLocation.Id into aBirthLocation
                            from BL in aBirthLocation.DefaultIfEmpty()
                            select new
                               {
                                   personnel.Id,
                                   personnel.FirstName,
                                   personnel.LastName,
                                   IssuanceLocation = IL.Name,
                                   BirthLocation = BL.Name
                               };
```

معادل است با:

```
SELECT
                        [Extent1].[Id] AS [Id],
                        [Extent1].[FirstName] AS [FirstName],
                        [Extent1].[LastName] AS [LastName],
                        [Extent2].[Name] AS [Name],
                        [Extent3].[Name] AS [Name1]
                        FROM [dbo].[People] AS [Extent1]
                        LEFT OUTER JOIN [dbo].[Places] AS [Extent2] ON [Extent1].[IssuanceLocationId] =
[Extent2].[Id]
                        INNER JOIN [dbo].[Places] AS [Extent3] ON [Extent1].[BirthPlaceId] =
[Extent3].[Id]
```

و البته این خروجی دوم فقط در صورتی تشکیل می‌شود که قسمت select new ذکر شود. در غیراینصورت مشکل select n+1 را پیدا می‌کند و اصلا چنین join ایی تشکیل نخواهد شد (در یک حلقه، به ازای هر شخص، یکبار کوئری select به جدول مکان‌ها تشکیل می‌شود). همچنین یک inner join هم علاوه بر left outer join تشکیل شده (برای فیلد غیرنال پذیر).

حتی همین حالت دوم را هم با کوئری ذیل که از خواص راهبری استفاده کرده، می‌توان تولید کرد:

```
var query = context.Personnel.Select(x => new
             {
              x.Id,
              x.FirstName,
              x.LastName,
              BirthPlaceName = x.BirthPlace.Name,
              IssuanceLocationName = x.IssuanceLocation == null ? "" : x.IssuanceLocation.Name
             });
```

با این خروجی SQL (به صورت خودکار برای فیلد نال پذیر، left outer join و برای غیر نال پذیر inner join تشکیل داده)

```
SELECT
    [Extent1].[Id] AS [Id],
    [Extent1].[FirstName] AS [FirstName],
    [Extent1].[LastName] AS [LastName],
    [Extent2].[Name] AS [Name],
    CASE WHEN ([Extent3].[Id] IS NULL) THEN N'' ELSE [Extent3].[Name] END AS [C1]
    FROM   [dbo].[People] AS [Extent1]
    INNER JOIN [dbo].[Places] AS [Extent2] ON [Extent1].[BirthPlaceId] = [Extent2].[Id]
    LEFT OUTER JOIN [dbo].[Places] AS [Extent3] ON [Extent1].[IssuanceLocationId] = [Extent3].[Id]
```

نویسنده: شاهد محمودی
تاریخ: ۱۳۹۲/۰۸/۲۲ ۲:۲۴

با تشکر اما جناب نصیری برای این مشکل من ، که هر 2 جدول اختیاری است و نال پذیر است فکر نکنم بشه از این روش استفاده کرد

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۸/۲۲ ۹:۱۸

فکر کنم خلاصه مطلبی که عنوان شده اینه که اگر در طراحی، فیلد نال پذیر *داشته باشید* ، در صورت استفاده از خواص راهبری متناظر با این فیلدها، به صورت خودکار left outer join درست میشه. بررسی‌اش هم نیازی به حدس و گمان نداره. مطلب لاگ کردن خروجی SQL می‌تونه کمک کنه .