

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 شبه 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 اين خروجى دوم فقط در صورتى تشكيل مى‌شود كه قسمت select new ذكر شود. در غير اينصورت مشكل [select n+1](#) را پيدا مى‌كند و اصلا چنين join ايبى تشكيل نخواهد شد (در يك حلقه، به ازاي هر شخص، يكبار كوئري select به جدول مكان‌ها تشكيل مى‌شود). همچنين يك inner join هم علاوه بر left outer join تشكيل شده (براي فيلد غيرنال پذير). حتى همين حالت دوم را هم با كوئري ذيل كه از خواص راهبري استفاده كرده، مى‌توان توليد كرد:

```
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 اين خروجى SQL (به صورت خودكار براي فيلد نال پذير، left outer join و براي غير نال پذير inner join تشكيل 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]
```

نويسنده: شاهد محمودى  
تاريخ: ۲۰۲۴/۱۳۹۲/۰۸/۲۲

با تشكر اما جناب نصيرى براي اين مشكل من، كه هر 2 جدول اختياري است و نال پذير است فكر نكنم بشه از اين روش استفاده كرد

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. بررسی اش هم نیازی به حدس و گمان نداره. [مطلب لاگ](#)  
[کردن خروجی SQL می تونه کمک کنه](#).