عنوان: بررسی خطای cycles or multiple cascade paths و یا cyclical reference در EF Code first
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۲۸ ۱۳:۵
آدرس: www.dotnettips.info
گروه‌ها: Entity framework

ابتدا مثال کامل این قسمت را با شرح زیر درنظر بگیرید؛ در اینجا هر کاربر، یک کارتابل می‌تواند داشته باشد (رابطه یک به صفر یا یک) و تعدادی سند منتسب به او (رابطه یک به چند). همچنین روابط بین کارتابل و اسناد نیز چند به چند است:

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Data.Entity;
using System.Data.Entity.Migrations;
using System.Data.Entity.ModelConfiguration;

namespace EF_General.Models.Ex18
{
    public class UserProfile
    {
        public int UserProfileId { set; get; }
        public string UserName { set; get; }

        [ForeignKey("CartableId")]
        public virtual Cartable Cartable { set; get; } // one-to-zero-or-one
        public int? CartableId { set; get; }

        public virtual ICollection<Doc> Docs { set; get; } // one-to-many
    }

    public class Doc
    {
        public int DocId { set; get; }
        public string Title { set; get; }
        public string Body { set; get; }

        [ForeignKey("UserProfileId")]
        public virtual UserProfile UserProfile { set; get; }
        public int UserProfileId { set; get; }

        public virtual ICollection<Cartable> Cartables { set; get; } // many-to-many
    }

    public class Cartable
    {
        public int CartableId { set; get; }

        [ForeignKey("UserProfileId")]
        public virtual UserProfile UserProfile { set; get; }
        public int UserProfileId { set; get; }

        public virtual ICollection<Doc> Docs { set; get; } // many-to-many
    }

    public class UserProfileMap : EntityTypeConfiguration<UserProfile>
    {
        public UserProfileMap()
        {
            this.HasOptional(x => x.Cartable)
                .WithRequired(x => x.UserProfile)
                .WillCascadeOnDelete();
        }
    }

    public class MyContext : DbContext
    {
        public DbSet<UserProfile> UserProfiles { get; set; }
        public DbSet<Doc> Docs { get; set; }
        public DbSet<Cartable> Cartables { get; set; }

        protected override void OnModelCreating(DbModelBuilder modelBuilder)
        {
            modelBuilder.Configurations.Add(new UserProfileMap());
            base.OnModelCreating(modelBuilder);
        }
    }

    public class Configuration : DbMigrationsConfiguration<MyContext>
```

```
    {
        public Configuration()
        {
            AutomaticMigrationsEnabled = true;
            AutomaticMigrationDataLossAllowed = true;
        }
    }

    public static class Test
    {
        public static void RunTests()
        {
            Database.SetInitializer(new MigrateDatabaseToLatestVersion<MyContext, Configuration>());
            using (var context = new MyContext())
            {
                var user = context.UserProfiles.Find(1);
                if (user != null)
                    Console.WriteLine(user.UserName);
            }
        }
    }
}
```

اگر این مثال را اجرا کنیم، به خطای ذیل برخواهیم خورد:

```
Introducing FOREIGN KEY constraint 'FK_DocCartables_Cartables_Cartable_CartableId'
on table 'DocCartables' may cause cycles or multiple cascade paths. Specify
ON DELETE NO ACTION or ON UPDATE NO ACTION, or modify other FOREIGN KEY constraints.
Could not create constraint. See previous errors.
```

علت اینجا است که EF به صورت پیش فرض ویژگی cascade delete را برای حالات many-to-many و یا کلیدهای خارجی غیرنال پذیر اعمال می‌کند.
این دو مورد در کلاس‌های Doc و Cartable با هم وجود دارند که در نهایت سبب بروز circular cascade delete (حذف آبشاری حلقوی) می‌شوند و بیشتر مشکل SQL Server است تا EF؛ از این لحاظ که SQL Server در این حالت نمی‌تواند در مورد نحوه حذف خودکار رکوردهای وابسته درست تصمیم‌گیری و عمل کند. برای رفع این مشکل تنها کافی است کلید خارجی تعریف شده در دو کلاس Doc و کارتابل را nullable تعریف کرد تا cascade delete اضافی پیش فرض را لغو کند:

```
public int? UserProfileId { set; get; }
```

راه دیگر، استفاده از تنظیمات Fluent و تنظیم WillCascadeOnDelete به false است که به صورت پیش فرض در حالات ذکر شده (روابط چند به چند و یا کلید خارجی غیرنال پذیر)، true است.

شبیه به همین خطا نیز زمانی رخ خواهد داد که در یک کلاس حداقل دو کلید خارجی تعریف شده باشند:

```
The referential relationship will result in a cyclical reference that is not allowed. [ Constraint name
=  ]
```

در اینجا نیز با نال پذیر تعریف کردن این کلیدهای خارجی، خطای cyclical reference برطرف خواهد شد.

نظرات خوانندگان

نویسنده: imo0
تاریخ: ۱۳۹۲/۰۷/۰۸ ۱۶:۲۲

سلام اقای نصیری . من دقیقا همین مشکل و خطا رو دارم با این تفاوت که نمیخوام CasCade delete رو غیر فعال کنم . در واقع من یه کلاس دارم که یکی از properties هاش یه لیستی از خود همین کلاس هست . یعنی به صورت تو در تو با خودش رابطه داره . مثله یه tree View . حالا من میخوام با حذف پدر تمام فرزندان و فزندان فرزندان و ... تا پایین همشون دیلیت بشن . یعنی همون on delete cascade خودمون . ولی خب اینجا میشه همونی که گفتین . حذف ابشاری حلقوی . این کار بخوام بکنم باید چیکار کنم . ؟ ممنون از لطفتون . منتظر پاسخم...

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۰۸ ۱۷:۳۴

مراجعه کنید به آخرین نظرات مطلب « مباحث تکمیلی مدل‌های خود ارجاع دهنده در EF Code first »

نویسنده: دانش پژوه
تاریخ: ۱۳۹۳/۱۰/۰۸ ۱۱:۱۶

با سلام
من با چنین مشکلی بر خوردم و به این آدرس رسیدم که از روش زیر استفاده می‌کرد:

```
modelBuilder.Entity<Employee>()
                    .HasRequired(e => e.SecondTeam)
                    .WithMany(t => t.SecondEmployees)
                    .HasForeignKey(e => e.FirstTeamId)
                    .WillCascadeOnDelete(false);
```

که متاسفانه متوجه نشدم. اگه زحمتی نبود در مورد این روش هم کمی توضیح بدید.
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۰۸ ۱۲:۵

در متن عنوان شد «راه دیگر، استفاده از تنظیمات Fluent و تنظیم WillCascadeOnDelete به **false** است که به صورت پیش فرض در حالات ذکر شده (روابط چند به چند و یا کلید خارجی غیرنال پذیر)، true است. » « از این لحاظ که SQL Server در این حالت (**true** بودن حذف آبشاری) نمی‌تواند در مورد نحوه حذف خودکار رکوردهای وابسته درست تصمیم‌گیری و عمل کند »