

第06期-隐藏属性与术语约定

2019年4月15日 23:22

隐藏属性

隐藏属性是指：在实体类中未定义，但在数据库中有字段定义。

Data Annotations

不支持通过数据注解的方式配置隐藏属性。

Fluent API

```
modelBuilder.Entity<Blog>().Property<DateTime>("LastUpdated");
```

赋值与查询

```
context.Entry(myBlog).Property("LastUpdated").CurrentValue = DateTime.Now;
```

```
var blogs = context.Blogs.OrderBy(b => EF.Property<DateTime>(b, "LastUpdated"));
```

数据库实体关系

关系定义了两个实体之间的关联，在关系数据库中，通过外键约束。

主要关系：一对多，一对一和多对多

相关术语：依赖实体、主体实体、外键、主键、导航属性（集合导航，引用导航，反转导航）

例如：Blog 和 Post 就是一对多关系，Post 是依赖实体，Blog 是主体实体，Post.BlogId 是外键，Blog.BlogId 是主体键，Post.Blog 是引用导航属性，Blog.Posts 是一个集合导航属性，Post.Blog 是 Blog.Posts 的反转导航属性。

按照约定，当发现类型上有导航属性时，将创建关系。如果属性指向的类型不能由当前的数据库提供程序映射为标量类型，则该属性视为一个导航属性。

约定

完全定义的关系

如果两个类型之间找到一对导航属性，则它们将被配置为同一关系的反转导航属性。

如果依赖实体属性包含名为<primary key property name>， <navigation property name> <primary key property name>或<principal entity name> <primary key property name>的属性，该属性将配置为外键。

```
public class Blog
{
    public int BlogId { get; set; }
    public string Url { get; set; }

    public List<Post> Posts { get; set; }
}

public class Post
{
    public int PostId { get; set; }
    public string Title { get; set; }
    public string Content { get; set; }

    public int BlogId { get; set; }
    public Blog Blog { get; set; }
}
```

没有外键属性

微软建议定义外键属性，但有时候确实不需要，将自动生成名为 <navigation property name> <principal key property name> 的隐藏属性。

```
public class Blog
{
    public int BlogId { get; set; }
    public string Url { get; set; }

    public List<Post> Posts { get; set; }
}

public class Post
{
    public int PostId { get; set; }
    public string Title { get; set; }
    public string Content { get; set; }

    public Blog Blog { get; set; }
}
```

```
}
```

单一导航属性

没有反导航属性和外键属性，通过约定定义一个关系。

```
public class Blog
{
    public int BlogId { get; set; }
    public string Url { get; set; }

    public List<Post> Posts { get; set; }
}
```

```
public class Post
{
    public int PostId { get; set; }
    public string Title { get; set; }
    public string Content { get; set; }
}
```

外键指定（数据注解方式）

```
public int BlogForeignKey { get; set; }
```

```
[ForeignKey("BlogForeignKey")]
public Blog Blog { get; set; }
```

外键指定（Fluent API）

```
modelBuilder.Entity<Post>().HasOne(p => p.Blog).WithMany(b =>
b.Posts).HasForeignKey(p => p.BlogForeignKey);
```