

第12期-表拆分与固有实体类型

2019年4月25日 17:04

表拆分

EF Core 中支持将两个或多个实体映射到单个表。这称为表拆分或表共享。

```
public class Order
{
    public int Id { get; set; }
    public OrderStatus Status { get; set; }
    public DetailedOrder DetailedOrder { get; set; }
}
```

```
public class DetailedOrder : Order
{
    public string BillingAddress { get; set; }
    public string ShippingAddress { get; set; }
}
```

```
modelBuilder.Entity<DetailedOrder>()
    .ToTable("Orders")
    .HasBaseType((string)null)
    .Ignore(o => o.DetailedOrder);
```

```
modelBuilder.Entity<Order>()
    .ToTable("Orders")
    .HasOne(o => o.DetailedOrder).WithOne()
    .HasForeignKey<Order>(o => o.Id);
```

固有的实体类型

显式配置

```
[Owned]
public class StreetAddress
{
    public string Street { get; set; }
    public string City { get; set; }
}

public class Order
{
    public int Id { get; set; }
    public StreetAddress ShippingAddress { get; set; }
}
```

```
modelBuilder.Entity<Order>().OwnsOne(p => p.ShippingAddress);
```

当 ShippingAddress 为私有属性时可通过字符串版本的重载函数指定。

```
modelBuilder.Entity<Order>().OwnsOne(typeof(StreetAddress), "ShippingAddress");
```

默认情况下固有属性名为: Navigation_OwnedEntityTypeProperty

```
modelBuilder.Entity<Order>().OwnsOne(
    o => o.ShippingAddress,
    sa =>
    {
        sa.Property(p => p.Street).HasColumnName("ShipsToStreet");
        sa.Property(p => p.City).HasColumnName("ShipsToCity");
    });
```

多个相同类型的固有属性

```
public class OrderDetails
{
    public DetailedOrder Order { get; set; }
    public StreetAddress BillingAddress { get; set; }
    public StreetAddress ShippingAddress { get; set; }
}
```

嵌套固有的类型

```
public class StreetAddress
{
    public string Street { get; set; }
    public string City { get; set; }
}

public class OrderDetails
{
    public StreetAddress BillingAddress { get; set; }
    public StreetAddress ShippingAddress { get; set; }
}

public class Order
{
    public int Id { get; set; }
    public OrderDetails OrderDetails { get; set; }
}
```

```
modelBuilder.Entity<Order>().OwnsOne(p => p.OrderDetails, od =>
```

```
{
    od.OwnsOne(c => c.BillingAddress);
    od.OwnsOne(c => c.ShippingAddress);
});
```

将固有类型映射到单独表中

```
modelBuilder.Entity<DetailedOrder>().OwnsOne(p => p.OrderDetails, od =>
{
    od.OwnsOne(c => c.BillingAddress);
    od.OwnsOne(c => c.ShippingAddress);
    od.ToTable("OrderDetails");
});
```

固有类型的集合

```
modelBuilder.Entity<Distributor>().OwnsMany(p => p.ShippingCenters, a =>
{
    a.HasForeignKey("DistributorId");
    a.Property<int>("Id");
    a.HasKey("DistributorId", "Id");
});
```

查询固有的类型

```
var order = context.DetailedOrders.First(o => o.Status == OrderStatus.Pending);
```