7.26 作业

Table per Type(TPT)继承 一个表对应一个类型  
Table per Class Hierarchy(TPH)继承 一个表对应一个类的层级(父类和子类的集合)  
Table per Concrete Class(TPC)继承 一个表对应一个实体类(忽略虚类)

下面贴上一些我查到的源代码简单的分析一下：

首先创建一个虚类person：

public abstract class Person //虚类

{

public int Id { get; set; }

[MaxLength(20, ErrorMessage = "名称长度不能大于8")]

public string Name { get; set; }

public string remark { get; set; }

}

[Table("Employee")]

public class Employee : Person

{

public decimal Salary { get; set; }

}

[Table("Vendor")]

public class Vendor : Person

{

public decimal HourlyRate { get; set; }

}

## 对于TPT类继承可以这么写：

class MyContext : DbContext

{

public virtual DbSet<Person> Peoples { get; set; }

public DbSet<Vendor> Vendors { get; set; }

public DbSet<Employee> Employees { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

string sqlServerConnStr = "xxx";

optionsBuilder.UseSqlServer(sqlServerConnStr);

base.OnConfiguring(optionsBuilder);

}

}

将会在数据库创建一张表 Peoples 包含列: Id,Name,remark,Salary,HourlyRate,Discriminator  
其中列Discriminator是EF自动产生用来区分Employee或者Vendor ;字段的取值也是Vendor或Employee

## TPC继承 DBContext如果这么写

class MyContext : DbContext

{

// public virtual DbSet<Person> Peoples { get; set; } //不要这句

public DbSet<Vendor> Vendors { get; set; }

public DbSet<Employee> Employees { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

string sqlServerConnStr = "xxx";

optionsBuilder.UseSqlServer(sqlServerConnStr);

base.OnConfiguring(optionsBuilder);

}

}

将会在数据库创建两张表Employee和Vendor包含列:

Employee: Id,Name,remark,Salary

Vendor: Id,Name,remark,HourlyRate

TPC继承产生的两张表的主键会重复,导致EF使用Person的时候产生主键冲突,所以不推荐使用这种方式

## TPT继承就是在数据库产生三张表