基于 NHibernate 的 ORM 映射机制研究

李斌勇,李 庆
(西南交通大学 CAD 工程中心,四川 成都 610031)

摘 要:针对面向对象设计与关系数据库设计之间的"阻抗不匹配"问题,提出了运用 ORM 技术来解决对象-关系映射冲突。深入研究对象/关系的映射机制,引出了基于.NET 的持久化框架 NHibernate,探讨了基于 NHibernate 技术的 ORM 映射机制,实现了对象和关系数据库之间的高效映射,达到了将业务逻辑层与数据存储有效分离。提出的多种映射策略将极大地缩短开发周期、降低系统开发成本。

关键词:对象/关系映射;关系数据库;关系模型;NHibernate
中图分类号:TP391 **文献标识码**:A **文章编号**:1673-629X(2009)07-0032-03

Research on ORM Mapping Mechanism Based on NHibernate

LI Bin-yong, LI Qing
(Engineering Center of Computer Aided Design, Southwest Jiaotong University, Chengdu 610031, China)

Abstract: Aimed at the "impedance mismatch" problem between the object-oriented design and the relational database design, proposes to solve the conflict of object - relational mapping by the technology of ORM. Studies deeply the O/R mapping mechanism, and describes the persistence framework—NHibernate which is based on .NET platform. Then researches on the ORM mapping mechanism based on the NHibernate and realizes the effective mapping of object and relational database, it achieves the effective separation of business logic layer and data storage. Presented a variety of mapping strategy will greatly shorten the software development cycle and reduce system development costs.

Key words: object/relational mapping; relational database; relational model; NHibernate

0 引 言

相对于 Java 平台下 ORM 技术的蓬勃发展,基于.NET 平台下的 ORM 技术还处在一个正在起步的阶段。对象-关系映射(Object/Relation Mapping,简称 ORM),是随着面向对象的开发方法和关系数据库的不断发展而产生的。ORM 主要实现程序对象到关系数据库数据的映射,它涉及到两个关键点:Object(对象)和 Relation(关系)。他们分别代表了每个应用系统要处理的两个核心工作:对对象的操作和对关系型数据库的访问[1]。在实际的软件开发中,开发人员也热衷于把这两种技术的结合应用作为首选,但面向对象技术是基于对象的相关理论,而关系数据库技术是基于关系理论,理论基础的不同直接导致了两种技术的"阻抗不匹配"。因此迫切需要一种解决不匹配的策略,而在面向.NET 环境中,一种对象/关系数据库映射(ORM)技术——NHibernate 是解决两者间不匹配的最有效方法。

收稿日期:2008-10-27;**修回日期**:2008-12-18
基金项目:国家 863/CIMS 高技术研究发展规划资助项目(2004AA414010)
作者简介:李斌勇(1982-),男,四川江油人,硕士研究生,研究方向为网络化协同电子商务。

1 NHibernate 体系结构

NHibernate 是基于 Microsoft.NET 的 O/R Mapping 持久化框架。它从数据库底层来持久化.Net 对象到关系型数据库,使应用程序代码仅仅和对象关联。它不仅管理.NET 类到数据库表的映射,还提供数据查询和获取数据的方法。NHibernate 在具体的操作业务对象时,不需要再写任何 SQL 语句,只需简单地操作对象的属性和方法,便可达到对数据库操作的目的。NHibernate 的体系结构如图 1 所示。

为了实现 NHibernate API 访问数据库,NHibernate 提供了五个核心接口:

●ISessionFactory 接口:初始化 NHibernate,充当数据存储源的代理,并创建 ISession 对象。

●ISession 接口:负责保存、更新、删除、加载和查

询对象。

●ITransaction 接口:负责管理事务,事务必须由 ISession 来启动。

●ICriteria 接口:是 Expression(表达式)数据加载接口,Expression 是一个关系表达式组合,通过它能产生 SQL 语句的 Where 部分,用户需要通过 ISession 来间接调用它。

●IQuer 接口:是 HQL 数据加载接口,HQL(Hibernate Query Language)是 NHibernate 专用的面向对象的数据查询语言,功能类似于数据库的 SQL 语言。

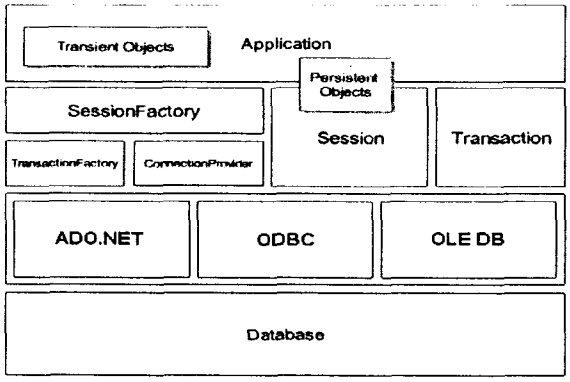

图 1　NHibernate 体系结构图

2　O/R Mapping 的实现原理

随着软件开发技术的发展,无论是.NET 还是 J2EE 下的高端应用程序开发,为了跨平台、跨数据库都普遍采用了 NHibernate 或 Hibernate 技术,为了使类(对象)和表相互的过渡,因此必须用最有效的 ORM 对象关系映射技术[2]。

ORM 的实现思想就是将关系数据库中表的数据映射成为对象,以对象的形式展现,这样开发人员就可以把对数据库的操作转化为对这些对象的操作[3]。因此,ORM 充当了连接应用程序和数据库的一个核心"桥梁"作用。如图 2 所示。

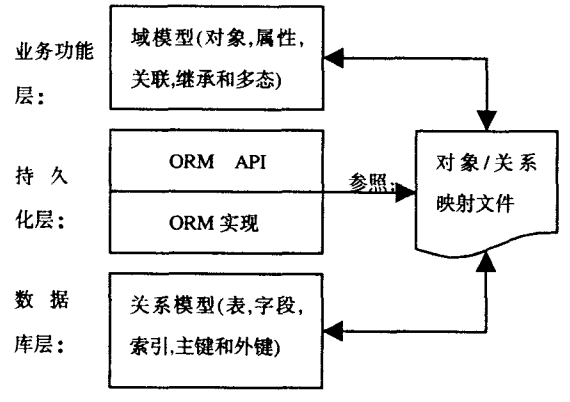

图 2　ORM 充当业务逻辑层和数据库层之间的桥梁

在利用 ORM 技术开发应用程序时,最核心的处理问题便是"映射"。为了成功实现对象与关系之间映射,必须建立的主要文件有:映射类(*.cs)、映射文件(*.hbm.xml)以及数据库配置文(*.cfg.xml)。

(1)映射类:作用是描述数据库表的结构,表中的字段在类中被描述成属性,将来就可以实现把表中的记录映射成为该类的对象。

(2)映射文件:作用是指定数据库表和映射类之间的关系,包括映射类和数据库表的对应关系、表字段和类属性类型的对应关系以及表字段和类属性名称的对应关系等。

(3)数据库配置文件:作用是指定与数据库连接时需要的连接信息,比如连接哪种数据库、登录用户名、登录密码以及连接字符串等。

在这三种主要的文件中,映射类为普通.NET 源文件、映射文件为 XML 格式、数据库配置文件为 XML 格式。要实现对象/关系间的"映射",必须要对以上三种文件进行解析。

3　O/R Mapping 映射机制的研究

虽然对象模型与关系模型是截然不同的数据模型,但在各自的领域内都有相应的概念与另一种数据模型的元素相对应[4]。选择有效的映射策略将对象模型向数据库关系模型映射,这就是类向数据库表的变换过程[5]。O/R Mapping 映射就是将对象模型与关系模型联系起来,由图 3 可以清晰地看出两者间的映射机制[6]。

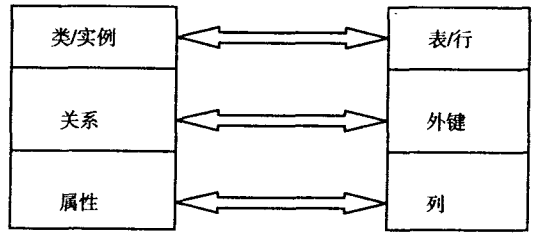

图 3　对象/关系数据映射机制

根据映射机制的不同,O/R Mapping 可归纳为实体映射和关系映射。

3.1　实体映射的策略

实体映射技术作为类与表之间的联系纽带,在 ORM 实现中起着至关重要的作用。根据抽象对象与关系数据库的特性,在 NHibernate 中,实体映射主要包括以下 3 部分内容。

3.1.1　类—表映射

在简单的情况下,一个类映射成一张表。但由于在面向对象思想中,类与类之间存在着复杂的继承关系。因此,不同层次的父类和子类映射到数据表时,应该根据实际的系统设计结果来权衡采用怎样的映射策

略,归纳后主要有三种基本解决办法:

(1)将整个类层次映射为单个数据表:这种方法需要将一个完整类层次结构映射成一个数据实体,而层次结构中所有类的属性都存储在这个实体中。

(2)将每个具体子类映射成单个数据表:使用这种方法,每个数据实体既包含自己的属性,又包含它所继承的父类的属性。

(3)将每个类均映射为数据库表:这种映射方法为每个类创建一张表,它的属性是 OID 和特定于该类的属性。

3.1.2 对象标识符(OID)—主键映射

应用程序按内存地址来标识或区别同一个类的不同对象,而关系数据库按主键值来标识或区别同一个表的不同记录。NHibernate 使用 OID 来统一两者之间的矛盾,OID 是关系数据库中的主键(通常为代理主键),等价于对象模型中的对象标识符。在运行时,NHibernate 根据 OID 来维持对象和数据库表中记录的对应关系。在设置主键时,为了避免因业务规则变化而引起的数据库字段和键值的频繁变化,除遗留系统外,对象的主键建议使用自动生成的无意义值,而不是具有实际意义的属性,比如人名、身份证号等。

3.1.3 属性—字段映射

类的属性对应数据库表中的列,一个属性可以与关系数据库中的一个或多个字段对应。故在把类映射到关系时,需要考虑类的属性类型。面向对象系统一般支持比关系模型更加丰富的类型系统,这时往往类的一个属性无法映射到一个关系的属性上,因此类的属性可能映射为多个关系属性,甚至在有些情况下映射为另外的关系(如数组等聚合类型)。

3.2 关联关系映射的策略

关联是对象之间最普遍的联系,描述给定类的对象之间语义上的连接,表达对象之间的连接数量关系。根据连接数量可以分为 one - to - one, one - to - many, many - to - many 三种映射[7]。

(1)one - to - one 映射:NHibernate 提供了两种一对一映射的方法,按主键映射和按惟一外键映射。

按主键映射:这种映射方式就是两张关联表通过共享主键形成一对一映射关系。只需为一张表设定主键生成器,而另一张表的主键与之共享相同的主键值。NHibernate 通过 one - to - one 节点进行声明。

例如对 Employee 表和 Person 表进行主键一对一关联的关键代码为:

```
<one-to-one name="Person" class="Person" />
<one-to-one name="Employee" class="Employee" constrained="true" />
```

按外键映射:这种映射方式就是将一张表的主键作为另一张表的唯一外键。NHbimeate 通过 many - to - one 节点进行声明。对于上面的 Employee 和 Person 的例子,如果使这种关联方式,应该表达成:

```
<many-to-one name="Person" class="Person" column="PERSON_ID" unique="true" />
```

如果在 Person 的映射加入下面几句,这种关联就是双向的:

```
<one-to-one name="Employee" class="Employee" property-ref="Person" />
```

(2) one - to - many 映射:实现一对多的映射与实现一对一的映射相似,所不同的是外键应放在对象关系"多"的一方对应的表中。一对多关系分为单向一对多关系和双向一对多关系。单向一对多关系只需在"一"方进行配置,双向一对多关系需要在关联双方均加以配置。

(3) many - to - many 映射:NHibernate 映射关系中相对比较特殊的就是多对多关联。多对多映射与一对一和一对多映射不同,它需要借助中间表完成多对多映射信息的保存[8]。关联表包含关系中涉及到多张表的主键,也就是每个表中某个对象的 OID,每条记录即这些 OID 的组合,表示哪些对象之间是存在关系。例如:学生与课程为多对多的关系,如图 4 所示,可按下述方式建立关系。

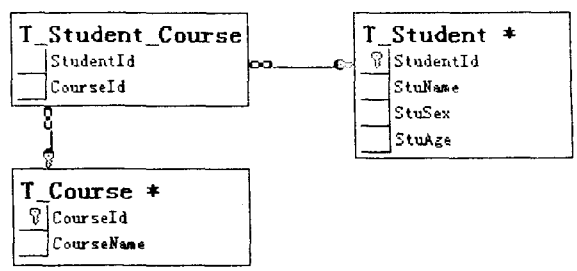

图 4 多对多关联的示例图

表 T_Course 映射文件的关键代码为:

```
<bag name="Students" table="T_Student_Course" lazy="true">
<key column="CourseId"/>
<many-to-many class="T_Student"
column="StudentId "/></bag>
```

表 T_Student 映射文件的关键代码为:

```
<bag name="Courses"
table="T_Student_Course" lazy="true"
inverse="true">
<key column="StudentId"/>
<many-to-many class="T_Course"
column="CourseId"/></bag>
```

(下转第 37 页)

并且计算 $\gamma(C,D,\beta) = 0.75$。然后根据给出的相对差异矩阵和 $\gamma(C,D,\beta)$ 的值逐步求出约简。最后求出属性约简的结果为 RED = $\{a_1,a_2,a_4\}$,而且 $\gamma(\text{red},D,\beta) = 0.75 = \gamma(C,D,\beta)$。

表 1　关于气象信息的决策系统

No	属性 C				分类 D
	Outlook(a1)	Temperature(a2)	Humidity(a3)	Windy(a4)	
1	Sunny	Hot	High	False	N
2	Sunny	Hot	High	True	N
3	Overcast	Hot	High	False	P
4	Rain	Mild	High	False	P
5	Rain	Cool	Normal	False	P
6	Rain	Cool	Normal	True	N
7	Overcast	Cool	Normal	True	P
8	Sunny	Mild	High	False	N
9	Sunny	Mild	Normal	False	P
10	Rain	Mild	Normal	False	P
11	Sunny	Mild	Normal	True	P
12	Overcast	Mild	High	True	P
13	Overcast	Hot	Normal	False	P
14	Rain	Mild	High	True	N
15	Sunny	Hot	High	True	P
16	Rain	Cool	Normal	False	N

根据给出的算法求出的属性约简是不是一个正确的约简,可以根据属性的重要度这个概念来进行检验。在参考文献[5]中给出定理:约简中的每一元对于约简中的其余的元都是重要的,约简集合外的每一元对于约简都是不重要的,重要度都为 0[5]。由这个定理,利用属性重要度公式就可以检验属性约简的结果。

3　结束语

属性约简是粗糙集理论中的一个重要的课题之一,但是信息系统的属性约简不唯一,要得到最简捷的决策规则必须得到最小约简[9]。但是要找出一个信息系统的最小约简已经证明是 NP - Hard 问题[10]。文中给出的算法通过实例证明可以有效地求出一个信息系统的 β 约简。而且给出利用属性重要度的概念来检验所求出的结果是不是正确的约简。

参考文献:

[1] Pawlak Z . Rough sets[J]. International journal of information and computer sciences,1982,11(5):341 - 356.

[2] Ziarko W. Variable precision rough set model[J]. Journal of computer and system science,1993,46(1):39 - 59.

[3] 张文修,吴伟志.粗糙集理论与方法[M].北京:科学出版社,2003.

[4] An A,Shan N,Chan C,et al. Discovering rules for water demand prediction: an enhanced rough set approach[J]. Engineering Application in Artificial Intelligence,1996,9(6):645 - 653.

[5] 史开泉,崔玉泉. S - 粗集与粗决策[M].北京:科学出版社,2006.

[6] 于兴刚.粗糙集属性约简算法在数据挖掘中的研究[D].重庆:重庆大学,2004.

[7] 于　冰.在数据挖掘中粗糙集信息约简算法的研究及应用[D].北京:中科院,2005.

[8] 夏春艳.基于粗集属性约简的数据挖掘技术的研究与应用[D].长春:长春大学,2006.

[9] 覃伟荣,秦亮曦.基于粗糙集理论的条件属性动态约简算法[J].计算机技术与发展,2008,18(8):23 - 25.

[10] 陶　志,许宝栋,汪定伟,等.基于可变精度粗糙集理论的粗糙规则的挖掘算法[J].信息与控制,2004,33(1):18 - 22.

(上接第 34 页)

4　结束语

在大中型应用程序里,无一例外地都会涉及到对象与关系数据库的处理。通过对 O/R Mapping 映射机制的深入研究,运用 NHibernate 技术将应用程序中的对象和数据库表之间建立适当的映射关系,有效地解决了面向对象设计与关系数据库设计之间的"阻抗不匹配"问题。同时,采用对象关系映射模式,使业务数据得以从业务逻辑中提取出来。随着面向对象和关系数据库的不断发展,O/R Mapping 技术将在软件开发过程中发挥日益重要的作用。

参考文献:

[1] Keller W. Persistence Options for Object - Oriented Programs Proceedings of OOP[M]. New York: Wiley Computer Publishing,2004.

[2] 孙卫琴.精通 Hiberuate:Java 对象持久化技术详解[M].北京:电子工业出版社,2005:5 - 39.

[3] 余俊新,孙　涌.J2EE 中对象关系映射的研究与实现[J].计算机技术与发展,2007,17(3):88 - 94.

[4] 林寒超,张南平. Hibernate 技术的研究[J].计算机技术与发展,2006,16(11):310 - 313 .

[5] 刘　伟.对象/关系映射在.NET 环境中的实现[D].长沙:中南大学,2007.

[6] 何　铮,陈志刚.对象/关系映射框架的研究与应用[J].计算机工程与应用,2003,39(26):188 - 189.

[7] 张淑全.基于 Hibernate 数据层设计模式的研究与实现[D].大连:大连海事大学,2007.

[8] Keller W. Object/Relational Access Layers[C]//Proceedings of the 3rd European Conference on Patterns Language of Programming and Computing. New York: Wiley Computer Publishing,1998.