

文章编号:1000-1972(2005)03-0073-06

UML 中的类模式在关系数据库中的映射及其实现

张虹, 郑会颂

(南京邮电大学 管理工程系, 江苏 南京 210003)

摘要:针对 UML(统一建模语言)中的类模式向关系数据库数据模式的转换过程,提出了一整套分析方法,包括类图到表的映射策略以及类图之间的几种关系在关系数据库中的实现,从而将 UML 技术与关系数据库技术相结合,方便了数据库设计。

关键词:统一建模语言;关系数据库;类;数据库设计;映射

中图分类号:TP311.13;TP301.2 **文献标识码:**B

The Mapping of Class Mode in the UML to the Relational Database and Its Realization

ZHANG Hong, ZHENG Hui-song

(Department of Management Engineering, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract: In order to transform the class mode in the UML to the data mode in the relational database, a set of integral methods is presented, including the mapping strategies of classes to tables and the realization of several relations between classes in the relational database. The combination of the UML with the relational database could facilitate the design of the relational database.

Key words: Unified modeling language; Relational database; Class; Database design; Mapping

1 引言

统一建模语言(Unified Modeling Language, UML),经过不断地修改、完善,已经趋于成熟,成为一种有效的面向对象的软件开发工具。UML作为一种提供图形化模型的可视化建模语言,可以使系统结构更为直观,易于理解。本文讨论如何把 UML 与关系数据库设计相结合,将数据库设计统一于面向对象的软件分析设计过程中,以提高系统开发的效率。这是因为,传统的 E-R 模型结构简单,一般只针对数据进行建模。随着数据库规模的扩大,简单的 E-R 模型结构无法清晰地分析和描述问题,导致系统开发难度系数增大。而采用 UML 的分析方法设计数据库,能够使数据库模型清晰易懂,能够更加清晰地反映系统结

构,易于开发,缩短了系统开发周期。另一方面,现在的开发环境大多是面向对象的,而存储机制往往是基于功能分解的关系型数据库,同时在 DBMS(Database Management System)支持的数据库模型中,关系型数据库是最普遍的,事实上目前较为流行的对象关系数据库模型也是关系数据库模型的一个扩展^[1],因而存在如何将面向对象分析设计思想与关系数据库模型设计相统一的问题。

我们知道,E-R 图只描述实体之间的关联关系,而 UML 对象之间的关系不仅仅是关联关系,还有泛化、组合和聚合等更为复杂的描述。由于 E-R 模型结构与关系型数据结构是同构的,所以上述统一建模过程的关键在于将更为复杂的 UML 数据结构如何转化为关系型数据结构。UML 在对系统数据进行逻辑建模时,一般采用类模式来实现。类模式是 UML 建模

的核心技术之一,对于设计数据库模型有着重要作用,数据库的逻辑视图可由 UML 的类图衍生。

目前已提出一些关于 UML 到关系数据库的部分转换机制^[2~4],本文主要系统地阐述了 UML 中类模式到关系数据库的映射转换,提出了一整套的通用分析方法,并加以应用。

2 UML 中类图的映射策略

类图是面向对象系统的建模中最常见的图之一。类图显示了一组类、接口、协作以及它们之间的关系,主要用于对系统静态设计视图建模^[5]。其中,类是面向对象系统组织结构的核心,表示被建模的应用领域中的离散概念,是具有相同结构、行为和关系的一组对象的描述符号。UML 中类图的映射主要是指对象标识、属性类型、类这 3 个方面的映射。

2.1 对象标识映射为主键

设计数据库模型,合理选择主键是一个关键的问题。一般定义主键可以有两种方法:

(1) 为每个类增加一个对象标识符(OID)属性,将其映射为数据库中相应类表的主键。参见图 1,其中 $\langle\langle pk \rangle\rangle$ (primary key) 表示主键。对象标志符作为单独属性,所占空间较小而且大小相同,从而大大简化了主键选择方案,使得数据库更新时不会产生完整性问题,同时方便了数据库操作。唯一的遗憾是对象标志符不具有实际意义,不能反映实际内涵。

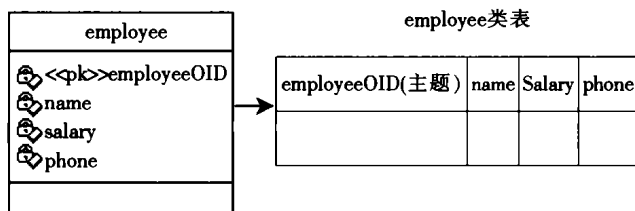


图 1 类的对象标识符映射为类表的主键

(2) 根据客观事实,将某个属性或属性的组合作为主键。该主键具有实际意义,容易进行维护;缺点在于涉及到外键,一旦其他类发生变化,更改比较困难。

2.2 属性类型映射为域

类的属性描述了其所有对象共有的特性。属性的类型可以是基本数据类型,如整数、实数、布尔型等,也可以是用户自定义类型。

属性类型对应于数据库中的域,域的使用可使数据库设计更具一致性,优化了数据库应用的移植性。一般来说,实现简单域比较方便,只须定义相应

的数据类型和空间大小。对于每个属性所映射的域,必须为该域的约束条件加入一条 SQL 语句,用以约束、检查域的取值。

2.3 类映射为表

通常,一个类映射为一张类表,类的属性映射为表的各列,类的对象则映射为表中的各个记录。值得注意的是存在以下两种特殊情况:(1) 类的属性中某些属性只是暂时性使用,不需要在数据库中永久保存,则该类属性无须映射。(2) 类的属性如果是多值,则该类属性映射为多个列^[4]。另外,由于附加对象标志符 OID 或附加关联关系等原因,需要在表中增加一些新的列。

3 UML 中类关系的映射策略

众所周知,类图由一系列类、接口和它们之间的关系(关联、泛化、聚集和组合等)所组成。其中泛化、聚集和组合具有单项导航性,即单一的方向性,而关联导航既可以单向,也可以双向,见图 2、图 3。由于关联导航的特殊性,可能在类图中会出现冗余。在设计数据库之前,我们可以对 UML 类图进行简化,去除一些冗余的关系。

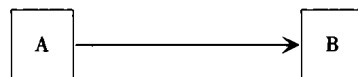


图 2 单向导航关联

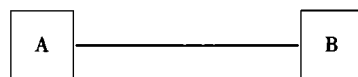


图 3 双向导航关联

举例来说,设 3 个类 A、B、C 唯一且不存在空值,同时它们之间的关系达到 BCNF(关系数据库中的 BC 范式,Boyce-Codd Normal Form)。AB 之间存在双向关联关系 R_1 , BC 类之间存在双向关联关系 R_2 , AC 之间存在双向关联关系 R_3 ,这 3 个关联所对应的实例分别为 r_1 、 r_2 、 r_3 ,则会出现如下情况:若 $r_3 = \prod_{R_3}(R_1 \bowtie R_2)$,那么 R_3 可以通过 R_1 、 R_2 推理得到,因而可以去掉 R_3 关系。将其归纳到一般情况:若类图中存在 n 个关系 $R_1, R_2, \dots, R_n (n \geq 2)$ 。假设 r_i 是 R_i 对应的关联实例($1 \leq i \leq n$)。如果存在任意 $i (1 \leq i \leq n)$,存在 $r_i = \prod_{R_i}(r_1 \bowtie \dots \bowtie r_{i-1} \bowtie r_{i+1} \times \dots \bowtie r_n)$,则 R_i 是冗余的关联关系,可以去掉。通过这种方法^[6]可以简化复杂的 UML 类图中

的一些关系,从而方便设计数据库。

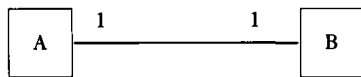
下面将一一阐述类之间的几种关系在数据库中的映射。

3.1 关联关系的映射

UML中的关联关系描述了系统中对象或实例之间的离散连接,是一种结构关系,规定了一种事物的对象可以与另一事物的对象相关联。当类参与关联时,类在关联关系中扮演一个特定的角色。

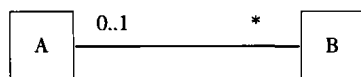
关联涉及的对象数目称为阶元,阶元的大小反映了关联的多重性,一般在关联端标出。常用的阶元标记有“0..1”、“1”、“*”等,分别表示0个或1个、1个、多个对象参与该关联。根据阶元的数目可将关联划分为4种^[7]:一对一关联,零或一对一关联,一对多关联和多对多关联。

(1) 一对一关联:

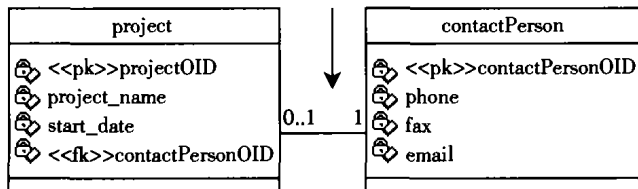


表示A的一个对象与B的一个对象关联。在这种情况下,可在两个类中任意选择一方,在其所对应的类表中添加一个外键 <<fk>> (foreign key), 指向另一方所对应类表中的主键,从而实现两张类表之间的连接,将关联关系成功映射到数据库中。需要注意的是,不要在两个表中均放置对方的主键,这样会造成冗余。

(2) 零或一对一关联:



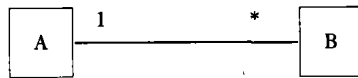
这表明B的一个对象可与0个或1个A对象发生关联,一般在A类(即对象个数为零或一的那一方)所对应的表中添加一个外键,指向另一方B类所对应的类表中的主键,建立两表之间的连接。此时,外键作用类似于面向对象技术中的指针,参见图4。



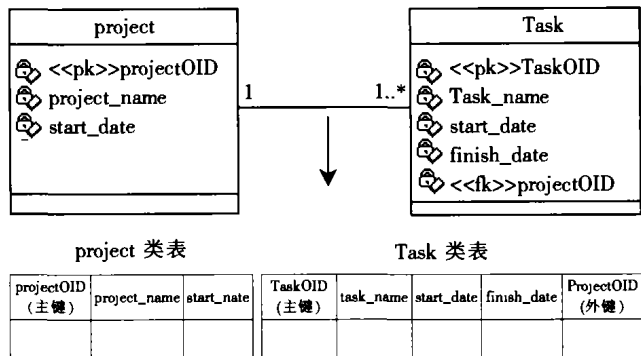
project 类表			contactPerson 类表			
projectOID (主键)	project_name	start_date	contactPersonOID (外键)	contactPersonOID (主键)	Phone	fax

图4 零或一对一关联

(3) 一对多关联:



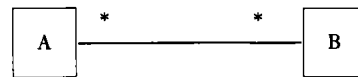
表示A的一个对象与B的多个对象关联,这种关联关系可以通过在B类(即具有多个对象的类)所对应的类表中增加一个外键,指向另一方A类的主键,从而建立两个表之间的关联,参见图5。



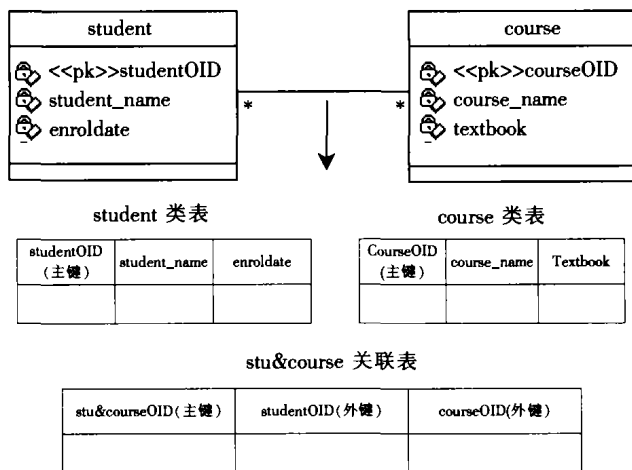
project 类表			Task 类表				
projectOID (主键)	project_name	start_date	TaskOID (主键)	task_name	start_date	finish_date	ProjectOID (外键)

图5 一对多关联的映射

(4) 多对多关联:



实现多对多关联,通常需要建立一个关联表,映射关联对象,从而将多对多关联转化为两个一对多关联。实现时,在新建的关联表中设置一个对象标志符OID,同时增加两个外键,分别指向初始关联的两个类对应表的主键,参见图6。



student 类表			course 类表		
studentOID (主键)	student_name	enroldate	CourseOID (主键)	course_name	Textbook

stu&course 关联表		
stu&courseOID (主键)	studentOID (外键)	courseOID (外键)

图6 多对多关联

3.2 泛化关系的映射

泛化关系是类元的一般描述和具体描述之间的关系,具体描述建立在一般描述的基础之上,并对其进行了扩展^[8]。从实现的角度来看,UML的泛化体

现了分类与继承原则,与面向对象程序设计语言中的继承性概念相关。一个子类继承超类的全部属性和方法,一个子类本身又可以有自己的子类,从而构成复杂的一般/特殊结构。

在关系数据库模型中没有直接的方法实现泛化,这里可以通过 3 种方法实现其映射^[4]。

(1) 类层次映射为单个表。通过这种方法,将泛化关系中所有类都映射在单个表中,所有类的属性都保存在该表中;同时在表中增加一个对象标志符 OID,以及一个对象类型,用以标识角色类型,如图 7 所示,这种映射方法支持多态,不足之处是当泛化关系中任一类发生变化时都会影响该表,导致耦合性的增加,同时占据了较大的存储空间。

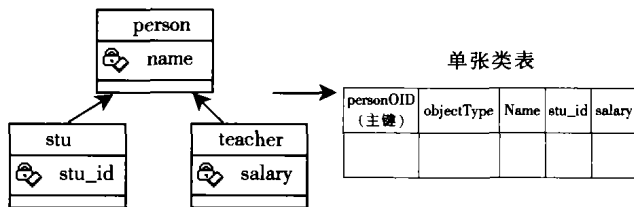


图 7 类层次结构映射为单张表

(2) 每个子类映射为单张表。将超类属性分别复制到各个子类中,这样每个子类既包含自身属性,又包含超类属性,同时在各个子类中增设各自的对象标志符 OID,从而实现映射。在这个过程中,超类不需要参与映射,如图 8 所示。

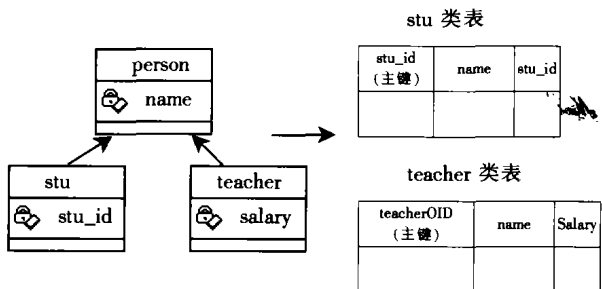


图 8 每个子类映射为单张表

由于每张表包含单一类的所有信息,故这种方法对于执行报表、专门报告比较方便。缺点是当修改超类时,须对所有子类都进行操作,同时当对象类型发生变化时,需要进行复制处理,比较繁琐。

(3) 每个类映射为单张表。为每个类都建一张表,但每张表中的对象标志符 OID 都设为超类的类表中的 OID。在子类的类表中,OID 既是主键又是外键,各自属性不变,如图 9 所示。

这种方法支持多态,较好地适应面向对象的要求,易于修改;但由于表数目较多,会造成访问数据库时间比较长。另外,每张表反映的不是单一类的

信息,所以不宜进行报表等工作。

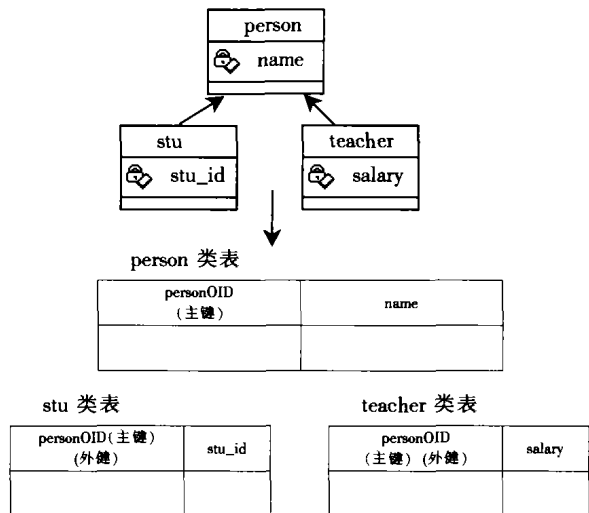


图 9 每个类映射为单张表

3.3 聚集关系的映射

聚集关系是一种特殊的关联关系,表示事物的整体/部分关系较弱的情况,代表“has-a”(拥有)关系。在这种情况下,“部分”能够为多个“整体”所共享,并且“部分”独立于“整体”而存在^[9]。将这种关系映射到关系数据库中,可分为两种情况:一种是聚集关系较为紧密的情况下,可将其映射在一张表中;另一种是聚集关系较为松散的情况下,可以用一对多关联的映射方法实现,须在子类的类表中增设一个外键指向超类的类表的主键。例如 office 类和 office Member 类之间存在着聚集关系,将该关系映射到关系数据库中,方法如图 10 所示。

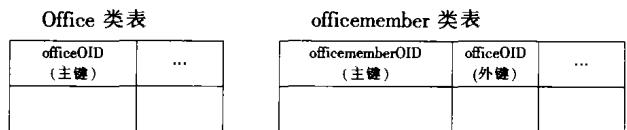


图 10 聚集的映射

3.4 组合关系的映射

UML 中组合关系是一种特殊的聚集关系,表示事物的整体/部分关系较强的情况,称为“contains-a”联系。此时,“部分”不能为多个“整体”所共享,只依附于一个“整体”而存在^[9]。例如,圆由点组成,圆和点之间就构成了组合关系。

具体的映射策略与聚集类似,由于组合关系中整体和部分间存在很强的所有关系和一致的生命周期,所以子类所对应的子表中的外键不能为空。

究,2001,(12):131~133.

- [5] BOOCH G, RUMBAUGH J, JACOBSON I. UML 用户指南[M]. 邵维忠, 麻志毅, 译. 北京: 机械工业出版社, 2001.
- [6] WAI Y M, PAPER D P. On transformations from UML models to object-relational databases[A]. Proceedings of the 34th Annual Hawaii International Conference on System Sciences[C]. 2001. 1194~1203.
- [7] 王黎明, 柴玉梅. UML 中的关联关系及其实现模式[J]. 郑州大学学报(理学版), 2002, (3): 25~28.
- [8] BOOCH G, RUMBAUGH J, JACOBSON I. UML 参考手册[M]. 姚淑珍, 唐发根, 译. 北京: 机械工业出版社, 2001.
- [9] PARDEDE E, RAHAYU J W, TANIAR D. Mapping Methods and Query for Aggregation and Association in Object-Relational Database using Collection[A]. Proceedings of the International Conference on Information Technology: Coding and Computing ITCC 2004[C]. 2004. 539~543.
- [10] 张龙祥. UML 与系统分析设计[M]. 北京: 人民邮电出版社, 2001.

作者简介:



张虹(1982-),女,江苏盐城人。南京邮电大学管理工程系硕士研究生。2003年毕业于南京邮电学院管理工程系。主要研究方向为信息系统与网络的管理。



郑会(1947-),男,广东潮阳人。南京邮电大学管理工程系副主任,教授。IEEE 会员、中国通信学会高级会员、中国电子学会高级会员、江苏省注册科技咨询专家。1982年毕业于南京邮电学院电信工程系。1993年在法国国立电信学院系统与网络系获硕士学位。目前研究方向为信息系统与网络的管理。

(责任编辑:胡长贵)

(上接第 72 页)

作者简介:



董传杰(1980-),男,山东莱州人。南京邮电大学计算机科学与技术系硕士研究生。2003年毕业于南京邮电学院计算机科学与技术系。目前主要研究方向是网格技术、移动代理和信息安全等。

王汝传(1943-),男,安徽合肥人。南京邮电大学计算机科学与技术系教授,博士生导师。(见本刊 2005 年第 1 期第 75 页)

(责任编辑:胡长贵)