一种通用的表结构设计方法①

A Generic Approach to Table Structrues Design

郑劲松　(浙江工业大学 机械制造及自动化教育部重点实验室 浙江 杭州 310014)

摘　要： 本文介绍了一种能适应多样化、复杂信息保存和处理要求的新型关系模式。把传统的关系模式分解为分别保存数据和操作这样两个数据结构，然后通过业务层代码对两者进行解释合并。这种处理方法彻底解决了关系模式对信息的依赖性，提高了关系模式的能力，也方便了应用程序的开发。

关键词： 关系数据库 关系模式 信息管理 关系模型 数据结构

在关系数据库理论中，关系模式定义为一个五元组：R(U , D , dom , F)。其中：R 代表关系名；U 代表从实体中抽象后，定义的一组属性；D 代表属性组 U 的取值域；dom 代表属性 U 到域 D 的映射；F 代表属性组 U 上的一组数据依赖[1]。从这个定义中，数据依赖规则说明关系模式的属性已经绑定了具体实体型信息。例如描述一组学生实体，通过抽象定义一个学生关系(学号，姓名，…)，属性学号就是学生实体的一个属性，学号的定义，包含了该属性的数据类型和操作属性。因此学生关系就被用来组织保存学生实体的信息和对于学生信息的各种操作。

但是在实际的信息管理项目中，我们经常会碰到为了保存信息类型或信息数量都不尽相同的这样一类实体信息，根据概念模型来设计，我们可以抽象出来很多的实体型，通常我们就给每一种实体型定义一个关系模式。这样我们的数据库就跟项目业务直接绑定了。由于业务变化是经常的，所以关系模式也要跟着改变才能适应业务管理需求，这样将导致上层应用软件的开发和维护变得非常困难。要解决这样的信息存储和管理问题，我们应该把关系模式和业务模式两者设计成间接相关，把关系模式定义为纯粹的数据结构，其本质就是一种分层体系思想的体现。

1　一种具有自适应性表结构

1.1 表结构设计

我们把关系模式定义成不绑定业务含义的朴素的关系模式 R(U)，其中 U 是数据项的集合；把属性的信息属性和操作属性，另外还有关系模式的域 D，映射 dom，规则 F 提到数据业务层上。因此，我们在数据库中定义的表结构如下：P (PK1,T01，T02，T03，T04,T05,…G01,G02,...)；其中：P 是关系表名，PK1 为关系表主码字段，T01,T02 等为保存字符型的字段，G01,G02 等为保存大对象型数据的字段。该关系表的字段数量可以根据要保存的数据项最大项数来定。所有字段没有具体含义，只是数据，数值、日期等常用简单类型数据都统一保存为字符类型格式，图片等大对象型数据用通用大对象型数据格式保存，尽量把各种数据类型转换成统一的数据格式保存。

下面为了描述的方便，我就在这里定义几个名词：小类和大类，数据表和信息表。我把概念模型中的实体型在这里称之为小类；把数据管理相似的一类实体型称之为一个大类。再来区别一下数据和信息的概念，数据仅仅是描述事物的一些符号，而信息则是赋予了具体含义的符号，在这里我们把存储数据的表称之为数据表，保存信息的表称之为信息表，保存数据的字段称为数据字段，保存信息的字段称为信息字段。所以上面定义的关系表 P 就是一张数据表，里面的字段 T 和 G 都是数据字段，在数据库层，他们仅仅被用来保存数据，数据的具体含义和他们的操作属性就被定义在上层业务解释层。这样我们就可以从一个数据表中派生出来很多信息表(如图 1)。但是这种体系不同于数据库的模式和外模式之间的映像，因为模式和外模

① 收稿时间:2008-09-25

式都属于数据库体系结构中，而我们这里的数据表存在于数据库中，而信息表在应用层，直接与应用界面逻辑相关的。另外，对于大类中的共性信息，我们可以直接在数据表中定义信息字段(如图 2)，这些字段可以不作解释直接应用在应用界面中。

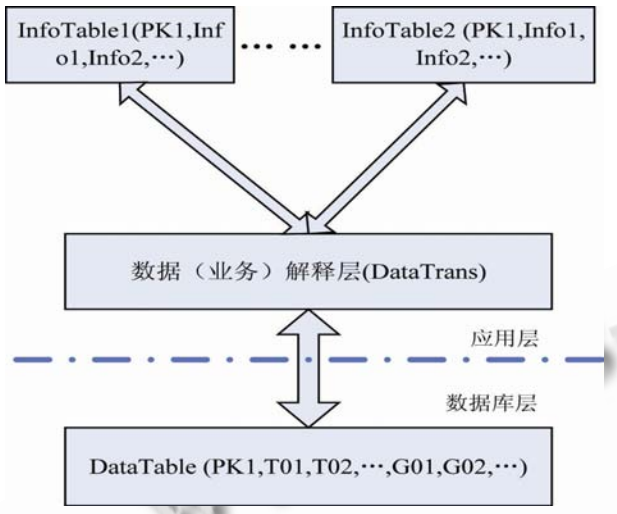

图 1　自适应表结构 1

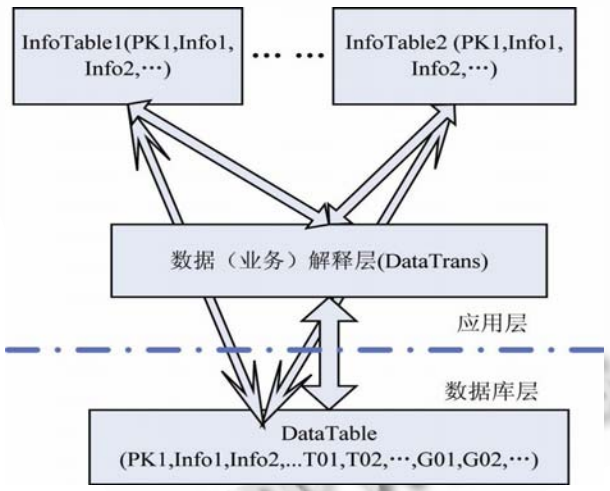

图 2　自适应表结构 2

对于业务解释层要用的的一些信息，我们也把他存入数据库中，保存业务信息的表结构设计如下：

B(EntryType，DataTableName，DataItemName，DataItemCaption，DataItemType，DefaultValue，CheckRule，DataIsNull，DataUnit，InputOrder)

EntryType-小类名称；DataTableName-数据表名；DataItemName-字段名；DataItemCaption-标题；DataItemType-数据类型；DefaultValue-默认值；CheckRule-信息有效性检查规则；DataIsNull-在界面上是否为必填信息；DataUnit-数据的单位；InputOrder-信息的录入顺序。

B 表中的每一行数据，就是对表 P 中每一个数据项的具体描述。如果数据项还有其他要描述的信息，只要在 B 表中增加一个属性。业务解释层根据 B 表中的描述，把 P 表中的数据和应用层的信息联系起来。

1.2 一种具有自适应性的主关键字

我们在设计关系模式的时候，每个关系模式一般都必须定义一个主键，通常情况下，我们选择能唯一标识实体的属性或属性组定义为主键。这样的话，主键就是具体的，是这类实体的主属性了，这个是通常的做法。现在，我们为了能适应多样性数据存储和处理要求，我们定义一个物理上没有具体含义的主键字段，在关系表中，该字段的值具有唯一性和非空性，字段的数据类型是字符型，定义足够存储长度。然后我们同样用一张业务表来解释这个主关键字段的业务含义。此业务表结构设计如下：

PK(EntryType,DataTable,PKItemName,SubItemType,PKSubItemCaption,CheckRule)

EntryType－小类名称；DataTableName－数据表名；PKItemName-主关键字字段名；SubItemType-关键字段子集类型；PKSubItemCaption-应用界面上显示的标题；CheckRule-信息有效性检查规则。

我们现在把物理关键字 PKItemName 字段拆分为几个域，每一个域解释为一个业务关键字段。例如：定义 PKItemName 字段的全面 10 个字符为委托单编号，后面 2 个字符为该委托单中试验序号。这样 PKItemName 字段就拆分成 2 个域，委托单编号的类型(SubItemType)我们定义为 A10，试验序号的类型定义为 B2。SubItemType 字段的值是用于拆分 PKItemName 字段的：格式为第一个字符为 A,B,C,…按顺序取其中一个字符，表示拆分顺序；后面为一个数值，表示拆分的位数。所有的拆分位数之和应该不大于 PKItemName 字段定义的字符长度。

2　小结

利用这样一种表结构定义和业务描述分开保存的方法，我们可以实现多样化、复杂的业务信息的存储和处理要求，真正达到业务管理软件的个性化定制。

参考文献

1 萨师煊,王珊.数据库系统概论.北京:高等教育出版社,2000.