

43-46

嵌入式关系数据库的查询语言

On the Query Languages for Nested Relational Database

褚东升

(山东财政学院信息管理系 济南 250014)

TP311.13

摘 要 The nested relational model has been proven useful in modeling databases of complex objects. In this paper we consider query languages designed specifically to exploit the power of this model. First, formal query languages are considered, next, two higher-level languages, SQL/NF and HDBL, are discussed and compared.

关键词 Nested relational database, Query languages

1 引言

嵌入式关系模型去掉了关系必须遵守第一范式的限制,不要求数据库中所有的域必须是原子的,所有非原子的域都被认为是值关系域(relation-valued),而结果则为关系中关系的嵌入。由于嵌入而产生的这种附加的建模能力允许表示更高层的用户接口和底层的实现方面的细节。在高层上,具有值集合(set-valued)域的“对象”可在单一的元组中而不是几个元组中表示之。在用户视图中元组和实体间相应关系导致了更为自然的接口。在底层,诸如分簇(clustering)或重复域这类的实现技术可以使用嵌入关系模型的形式来表示。

本文中我们所讨论的大多数面向用户的语言是基于 SQL 语言的,因为 SQL 现已作为一种标准的数据库查询语言而被广泛地接受并应用之。根据在 SQL 中融合进去嵌入关系的概念,很多研究者已经证明了这些概念很容易在实际查询中得到应用。事实上,在 SQL 中结合嵌入后可消除标准 SQL 中的若干“丑陋”的特性(或仅保持其向下兼容的目的)。

2 形式查询语言

下面我们对最近在嵌入关系模型的形式查询语言领域中所做的工作进行概括地小结。最初在这一领域所做的工作主要是集中在系统地阐述扩充的关系代数,而最近的研究工作则是集中在扩充关系演算上。

2.1 关系代数

文[1-2]报导了对关系代数进行扩充以适应于

嵌入式关系模型方面所做的研究工作。对关系代数的基本扩充包含以下运算符:

- (1) 传统的关系运算符扩充到嵌入关系:并($\cup$)、差($-$)、选择(σ)、投影(π)和笛卡尔乘积($\times$);
- (2) 两个重构运算符:嵌入(ν)和解除嵌入(μ)。

解除嵌入运算符把一关系变换成这样一种关系,即把被解除嵌入的关系中的每一元组连接到关系中其余的属性上以减少嵌入深度的这样一种关系。嵌入运算符则是根据等价类结构创建划分。两个元组为等价的,是指对于没有被嵌入的属性来说具有相同的值。被嵌入的属性构成了包含属性等价类中全部元组的嵌入关系。

所有其它已提出的扩充运算符均可根据上面所述的基本运算符表示之。Van Gucht^[3]证明了这些基本代数在 Bancilhon 和 Patadeans 的意义下是完备的。下面我们将介绍某些重要扩充的关系代数运算符及其相应的性质。

结构保持运算符,因为嵌入关系能被构造成为其所嵌入的模式中的所隐含的多值依赖进行保持,而保持这些性质的运算符是十分有用的。对并、差、投影和联结以及嵌入现在也已提出了某些扩充。

嵌入选择:在 XERSO 模型中, S. Abiteboul 和 N. Bidot 等人提出了一种具有可从关系中以及嵌入部件中同时执行选择功能的扩充运算符。

统计运算符:SSDB 系统中可支持统计数据库的运算符与嵌入关系数据库中的运算符具有相同的功能。

最近,我们注意到对于嵌入关系模型的递归代数问题上也已提出了两个方案。在这些代数中,运算

符在选择谓词和投影属性列表中可递归出现。这些代数可比非递归代数以更为自然的方式表示复杂查询。有人已经论述了查询变换和优化^[4]作为一种中间语言很有前途,但却不能提供任何更强的表达能力。

2.2 关系演算

已经证明,查询语言的最好形式为基于谓词演算的非过程语言,因为它具有较强的表达能力。然而,最初的嵌入关系语言的研究工作则是集中在代数(过程)语言上而并非是集中在嵌入关系模型的演算语言上的。Roth^[5]是第一个提出扩充关系演算的人。在 Roth 所提出扩充的演算中仅包含两个对原始关系演算的基本扩充,即两个新原子。一个用来说明联系的“element of”,另一个允许在简单原子中嵌入演算表达式。另外,这种扩充还在扩充演算表达式的类型上使用了语义约束来消除“不安全”表达式。所谓“不安全”表达式即为那些可产生无限或募集的关系,或者其计算时间为无限的表达式。使用这些限制,Roth 证明了扩充的代数与扩充的演算在其表达能力上是等价的。

文[2]中提出了另一种方案,该方案考虑值集合(set-valued)属性和聚集(aggregate)函数。该方法仅考虑了嵌入深度为一层的嵌入关系。文[2]中使用了 Klug 的方法证明了具有一层嵌入和聚集函数的模型的代数与演算是等价的。

3 类 SQL 查询语言

对于数据库系统现实世界的用户来说,简洁的代数和演算语言使用起来太困难且不好掌握。为了满足这类用户,在现有的关系数据库管理系统中定义并使用了若干种所谓“糖衣句法”的演算语言。在这里我们把重点放在对 SQL 语言的扩充和改编上,因为 SQL 语言现已发展成为关系数据库查询语言的工业标准。首先我们介绍对该语言的一种称为 SQL/NF^[6]的基本扩充。这种扩充后的语言是以一种与设计无关的方式对嵌入关系进行操作的。然后我们将介绍一种为 AIM-P 开发而提出的一种语言,该系统在关系及类 SQL 语言中增加了一些新的数据类型以对其语言进行更有效的支持。本节最后将讨论对以上所做工作的两种扩充,即属性作用和递归查询的使用。

3.1 SQL/NF

在 SQL 中,基本的查询机构为:

```
SELECT attribute-list
```

```
FROM relation-list
WHERE predicate
```

这种 SFW 表达式可按对 relation-list 中的全部关系构造一笛卡尔积的形式概念地执行之,在该笛卡尔积中选择出仅满足 predicate 的元组,然后在 attribute-list 中选择那些出现的属性。若不需要对元组进行限定,则 WHERE 子句可以省略。在传统的数据库中,每个关系均严格地由标量构成。在嵌入关系数据库中,每个关系可包含其它关系,也可以包含标量。在嵌入关系数据库结构定义中有效地使用了不相关原则。只要传统关系中可以出现标量的地方,我们均可以构成其关系。在 SQL/NF 语言中也使用了一简单变换。SQL 具有闭包特性,即在一个或多个关系上的任何一个查询的结果其本身亦可为关系。不相关原则则允许在任何出现关系名的地方使用 SFW 表达式。这样,SQL/NF 允许在 FROM 子句和 SELECT 子句中出现 SFW 表达式。对 SELECT 子句的修改显然是必不可少的,这样就可利用 SQL 的全部功能对嵌入关系进行存取。

SQL/NF 在设计中也试图消除 SQL 中的某些限制并尽可能地简化语法。如允许查询表达式出现在 FROM 子句中(在视图机制中允许非直接地出现)以及使用更简单的、更强有力的聚集函数的说明。另外,GROUP BY 机制可由 NEST 操作所取代并且在 SELECT 子句中使用 SFW 表达式的能力可进一步简化用户所需处理的构造集合。

3.2 HDBL

HDBL^[7]是 IBM 德国 Heidelberg 科学中心开发的 AIM-P 中使用的一种类 SQL 数据库语言。该语言严格遵循文[8]中的建议,其原型为支持嵌入关系模型、有序嵌入关系、时间版本数据、正文数据、长域以及向量及矩阵的大型项目。HDBL 不仅具有 SQL/NF 的很多基本特征,而且还有对序关系、序表以及多重表的操纵。HDBL 还有一些涉及到 SFW 表达式如何构造关系的不同的假设。下面我们对各个语言的特性进行比较。

3.3 SQL/NF 与 HDBL 比较

图 1 分别给出了传统数据模型、基于 SQL/NF 的嵌入关系模型以及基于 HDBL 的扩充嵌入关系模型的比较。因为 SQL/NF 仅允许关系中的关系,故查询关系需要一个更为简单的构造集合。每一查询产生一个具有 SELECT 子句中所列出的属性的关系,其结果总是元组的集合。在 HDBL 中则不是这样,集合构造具有四种可能。HDBL 具有多种特性

与其扩充模型有关,这就使得 SQL/NF 无法与之相比。

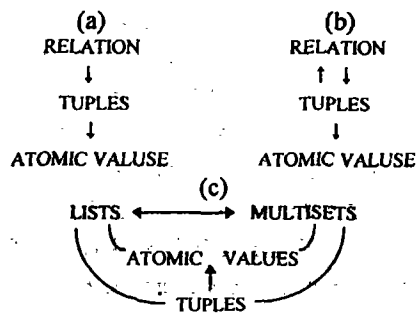


图1 不同模型的结构概念
(a)传统关系模型 (b)NF² 模型
(c)扩充 NF² 模型

有很多与扩充模型的子集相关的特性,这些特性相应于一可比较的基嵌入关系模型。令人感兴趣的工作是对 HDBL 的扩充以简化嵌入模式的投影运算。要说明的是,SQL/NF 和 HDBL 采用不同的方法在嵌入关系中进行插入、修改和删除元组。当在传统关系模型中表达式的 SELECT 子句中指出投影的属性时,是以一种结果关系的形式给出一列表,对于嵌入关系模型也是同样的。SQL/NF 查询在结构上相同,在 SELECT 子句中使用嵌入表达式来从嵌入关系中选择属性。HDBL 方法是简单地指出所涉及到的属性并且给出一诸如结果形式的“样板”,SQL/NF 和 HDBL 采用了十分不同的方法定义这些运算。HDBL 提出了一组合的方法,该方法使用一种对某些运算的传统程序设计的命令形式操作和一说明形式来保持其它操作的类 SQL/NF 结构。另一方面,SQL/NF 提出了所有的操作均在说明形式中进行。

3.4 对作用联结的扩充

下面我们介绍一种嵌入关系语言 SQL/NF 的作用扩充版本 X-SQL/NF。通常一数据库模式中的实体可有多种作用,如一个人可以是一名学生、雇员、经理等。根据作用唯一性假设而设计的数据库模式则需要其每个属性准确地与作用相对应。在大多数传统的数据库中,“ISA”关系在设计中不能明显地表示出来,这样,在属性间不存在系统层上的连接来表示人、学生、雇员或经理,即使他们是同一实体集的作用。对用户来说,要想从数据库中获得准确的信息则需要大量的知识。X-SQL/NF 中的 SQL/NF 有两个主要的句法增强部分:

- 定义作用(define roles)语句,该语句可允许

在属性中定义 ISA 联系;

• 如同在 SQL/NF 和 SQL 中使用“.”记号表示有关嵌入子关系或元组域一样,在 X-SQL/NF 中也使用“.”记号来表示“property of”。

为了说明 X-SQL/NF 的功能,我们考虑一个使用 SQL/NF 数据定义语言所定义的一个样本嵌入关系数据库:

```
manager = (mname, rank, dname, employee)
employee = (ename, salary, dependent)
dependent = (dname, dssn)
person = (name, ssn, b-day)
b-day = (month, day, year)
department = (dname, employee)
```

在 X-SQL/NF 中我们使用作用定义语句对作用信息进行如下说明:

```
define roles
mname ISA ename
ename ISA name
dname ISA name
dssn ISA ssn
```

虽然作用定义语句很简单,但“.”运算符的过多使用却导致了语义的复杂化。凡是在 SQL/NF 中属性名可以出现的任何地方也同样允许“.”表达式出现。

表达式 A.B 的意义为:(1)关系 A 上元组的 B 域;(2)嵌入在一关系 A 元组中的 B 关系;(3)一元组 A 域的 B 性质(隐含定义在一关系上的)。

要说明的是,从用户的观点来看,并不希望出现这种复杂性。然而在文[4]中论述的“.”运算符的过多使用实际上是简化了语义。例如,考虑表达式 manager.salary,这里有一带 salary 属性(见上面情况 1)的关系 manager。然而也可能 salary 是出现在 manager 关系的嵌入子关系中的属性,因为若不产生歧义性的结果时(以上情况 2),如 SQL/NF 这样的语言允许用户省略 manager 嵌入子关系的名字。manager 也可扮演 employee 的作用及另一关系。若想获得 manager 的薪水(以上情况 3),我们必须使用 department(employee, salary)关系。三种情况的使用不会改变其表达式在用户层上的语义,系统获得信息的方法依赖于低层的数据库模式,但值得注意的是,这些差别对用户来说并不重要,重要的语义问题是“manager 的 salary”是指付给 manager 的总费用,而并非是指给 manager 所管理的全体雇员的费用。虽然在 X-SQL/NF 中仅能对语义进行隐式定义,但这种情况并不比在标准关系语言中更糟。

3.5 递归关系的扩充

下面我们概述一下 Linnemann 在对 HDBL 进行扩充使其具有递归查询机制上所做的工作。我们

使用图 2 的例子来说明这种方法。

station	{destinations}
Frankfurt	Stuttgart, Munich
Stuttgart	Munich, Frankfurt
Munich	Frankfurt, Stuttgart, Salzburg
Salzburg	Munich, Wien
Wien	Salzburg

图 2 火车关系

[例] 考虑一火车关系连接数据库,对每一站示出了一直接可达的目标站的集合。图 2 示出了火车的例子。

HDBL 可通过一固定数据的中转路线到达某些城市,但是要想计算出所有可变中转路线,若没有递归机制是无法解决的。下面为一种可能的方法:

```
SELECT [station:r.station,
        destinations:r.destinations UNION
(SELECT d
  FROM t IN trains, d IN destinations
   WHERE EXISTS c IN destinations;
    c=t.station)
FROM r IN trains
```

目的地的表达式是一个具有左端目的地的等式,该等式的解可使用一个循环使其不断添加进去目的城市(在内层 SFW 表达式执行)直到其目的地不在发生变化来计算之。对于图 2 所示的关系其计算结果如图 3 所示。要说明的是,相应于传统的关系模型的查询构造是十分困难的。

station	{destinations}
Frankfurt	Stuttgart, Munich, Frankfurt, Salzburg, Wien
Stuttgart	Munich, Frankfurt, Stuttgart, Salzburg
Munich	Frankfurt, Stuttgart, Salzburg, Munich, Wien
Salzburg	Munich, Wien, Frankfurt, Stuttgart, Salzburg
Wien	Salzburg, Munich, Wien, Stuttgart, Frankfurt

图 3 火车关系的递归查询结果

这主要是因为缺少嵌入集属性的缘故,这对用户以及递归查询优化来说也是十分麻烦的。

结束语 在一程序设计语言接口中使用 SQL/NF 或其它关系语言的困难所在是因为典型的程序语言是面向记录的,而关系语言则是面向集合的。目前各种 4GL 对于非嵌入关系模型已做了尝试以解决这一问题,通常使用的方法是迭代构造。递归 HDBL 可以认为是等价于 4GL 的嵌入关系,通过递归可达到其处理能力。递归机制是存取递归嵌入式关

系数据库的一种自然方法,且与迭代具有相同的计算能力。在嵌入关系语言中引入递归可使其与如 Datalog 这类逻辑查询语言相比美。

目前嵌入关系查询语言的设计仍为一十分活跃的领域。本文中所介绍的语言虽然还有许多进一步的工作所做,但仍可充分说明嵌入关系模型的潜力。目前在该研究领域一个十分重要的问题是对嵌入关系的查询处理理论的扩充。由于嵌入关系表达高层用户接口及低层实现策略,而这种理论在实际应用中将显得十分重要。查询优化的问题对于嵌入关系语言的递归变式也是特别有意义的,最近对逻辑查询处理的研究或许可影响到该问题的解决。

致谢: 本文得到了聂培尧教授的帮助,特此表示感谢!

参考文献

- [1] Nicole Bidoit, The VERSO algebra or how to answer queries with fewer joins, J. of Computer and System Sciences, 5(3)1987
- [2] Gulter Rin Ozsoyoglu et al., Extending relational algebra and aggregate functions, ACM Trans. on Database Systems, 12(4)1987
- [3] Derk Van Gucht, On the expressive power of the extended relational algebra for the unnormalized relational model, In Proc. of the Sixth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, 1987
- [4] Srinivasen Ramakrishnan, Design and implementation of a translator for SQL/NF with role joins, The Univ. of Texas at Austin, 1986
- [5] Mark A. Roth et al., Extended algebra and calculus for $\rightarrow$ 1NF relational databases, ACM Trans. on Database Systems, 13(4), 1988
- [6] Mark A. Roth et al., SQL/NF: A query language for $\rightarrow$ 1NF relational databases, Information Systems, 12(1)1987
- [7] F. Anderson et al., Advanced information management prototype (AIMP)-User manual of the on-line interface of the heidelberg database language prototype implementation, Technical Note TN86-01, Heidelberg Scientific Center, 1986
- [8] P. Pistor and F. Anderson, Design a generalized NF² model with an SQL-type language interface, In Proc. of 12th Intl. Conf. on VLDB, 1986