支持多介质数据库的NF²方法*)

田沧海　林钧海　（210016南京航空学院计算机系）

摘　要

由于NF² (Non-First Normal-Form) 理论的实质在于允许关系的属性是另外一个关系，本文提出了用规范关系存贮各种介质的基本数据并用NF²的嵌套结构构造复杂的多介质数据的方法，从而有效地支持了多介质数据库的管理。

一、引言

近年来，非第一范式NF² (Non-First Normal-Form)理论的巨大进展[1,2]和应用，使数据库系统不仅能够支持商业领域中使用的简单表格，而且能够支持大而复杂的结构对象，包括文字、图形、图像及声音数据等等，从而能有效地利用数据库在工程领域和其它领域中实现管理。一般认为，NF²数据模型是关系和层次模型的高度统一。由于该模型可以达到外模式、概念模式和内模式的高度一致，保证数据存取的有效性和完整性，所以它优于扩充的关系数据模型[3]；也由于NF²模型易于实现，有较为成熟的理论基础，并且还能够描述抽象的语义[5]，所以亦优于面向对象的数据模型[4]。

NF² 理论的本质在于允许关系的属性是另外一个关系，从而可以描述复杂的结构。为了支持各种介质数据(如数字/字符、文字、图形、图像、声音等)，我们用规范关系描述和存贮各种介质的基本数据[6]，如一段文字、一个图形等，通过NF² 构造生成各种应用，形成复杂的多介质数据。我们的实践表明，NF² 方法支持多介质数据库是可行的。本文最后介绍了我们待进一步开展的工作。

二、基本数据存贮

各种介质的基本数据可以通过关系数据库加以存贮，例如，一段文字可以用(行号，内容)这两个属性值来描述，一个圆可以用(圆的名字，半径）来描述等等。为了能够统一地对所有介质的基本数据进行描述，我们给出关系数据字典和属性数据字典的描述形式[7]：

可以看出，这种描述提供了一种完整的、一致的、无冗余的表示，可以统一存贮各种介质的基本数据：

1. 数字/字符存贮：最基本的数据存贮

SRI International, Oct,1983.

[2] Clocksin, W.F. Implementation Techniques for PROLOG Database, SOFTWARE-PRACTICE and EXPERIENCE, Vol 15(7), PP.669—675(July 1985)

[3] Kevin A.Buettner, Fast Decompilation of Compiled Prolog Clauses,Logic Programming Research Group, School of Computer & Information Science, Syracuse University.

[4] 李良良，一种顺序PROLOG机系统结构的研究，国防科技大学计算机系，博士论文，1987(12)

[5] 车敦仁，编译型PROLOG数据库的实现技术的研究，国防科技大学计算机系，硕士论文，1988

[6] GKD-PROLOG/VAX780用户手册，国防科技大学，人工智能研究室，1985,12

*) 航空航天部科学基金资助课题

表1 关系数据字典rel-dict

描述项	含义
关系名	记录用户建立关系时所给的名字
元组长度	指出存放此关系中每个元组所需字节数
属性个数	指出每个元组含有几个属性
关系类型	指出此关系存贮的介质类型，包括Table、Text、Graph、Image等
属性连接指针	指出此关系所含属性在属性字典中的位置

表2 属性数据字典attr-dict

描述项	含义
属性名	记录用户建立关系时同时给出的属性名字
属性长度	指出存放此属性所需字节数
属性偏移量	给出此属性相对于整个元组存放地址的相对偏移量
属性类型	属性所允许有的类型，包括Smallint，Int，Float，Char，Date等

形式，不再重述。

2. 文字存贮：用户可以定义文字存放时行的最大长度，通过（行号，内容）两个属性可定义文字的存放：

Create Text t_1(rownum smallint,content char(80))

其对应的数据字典描述如图1所示。

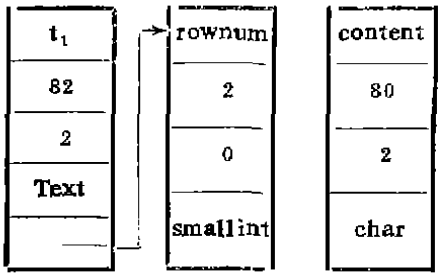

图1 文字t_1的存贮描述

因此，对于文字t_i的输入，每一行（rownum)最多可存贮80个字符的内容。

3. 图形存贮：基本图形分为规则图形和不规则图形，规则图形可以通过属性描述出来，例如Circular(name，radius)表示圆的信息（图2)。

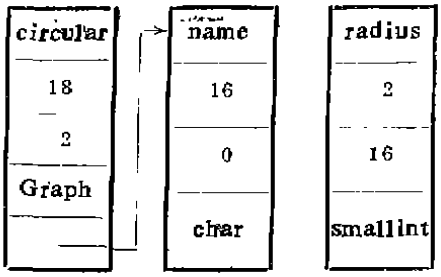

图2 圆的存贮描述

用户定义的所有圆均存贮在同一个关系中，通过圆的名字可进行存取，这样对于有限的规则图形，可以建立有限的不同关系加以描述和存贮。

对于不规则图形，我们引进“过程”的概念加以解决，任何用户需要的不规则图形均对应一个可执行程序(即过程)，执行的结果即相应的不规则图形，过程Procedure (name)的描述如图3所示。

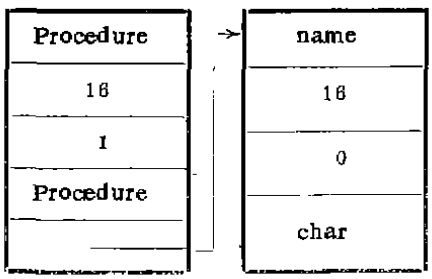

图3 过程的存贮描述

4. 图像存贮：可以用图像名（横座标，纵座标，颜色）来表示图像，也可以用一个“过程”来表示一幅图象。

5. 声音存贮：存贮频率、持续时间等属性值或用“过程”来表示。

三、NF^2方法

在上一节，我们论述了用规范关系可以描述和存贮各种介质的基本数据，由于NF^2数据模型允许关系的属性是另外一个关系，通过对规范关系的嵌套构造，可以描述和定义复杂的包括各种介质的数据，从而支持各种应用。为此，我们给出嵌套关系数据字典和属性数据字典的描述形式，即NF^2构造方法。

表3 嵌套关系数据字典nrel-dict

描述项	含义
嵌套关系名	记录用户定义嵌套结构时给出的名字
属性/子类个数	指出此嵌套关系的属性/子类个数
嵌套关系类型	指出此嵌套结构用来表示何种抽象语义，包括 Generation（概括）、Aggregation（聚集）等
属性/子类连接指针	指出此嵌套关系所含属性/子类在属性字典中的位置

表4 属性(子关系/子类)数据字典nattr-dict

描述项	含义
子关系/子类名	记录用户定义嵌套结构时同时给出的子关系/子类的名字
属性/子类个数	对于规范子关系或子类，此项值为0
子关系/子类的类型	对于嵌套子关系，类型为Ntable，而对于规范子关系或子类，类型为Table、Text、Graph、Image、Voice 等
左上角座标	记为 (x, y)，实际存放值i=(x≪8)+y
右上角座标	记为 (x, y)，实际存放值i=(x≪8)+y

表4中，左上角座标和右下角座标用来指定各介质基本数据的显示区域，通过各种基本数据的混合定位输出，可以满足用户的不同需求。

大部分系统如AIM[8]均利用NF^2数据模型直接支持嵌套表格，但需要数据是格式化的，不能支持非格式化数据的嵌套。我们的方法是利用NF^2的嵌套结构构造复杂的非格式化数据。

例如用户需要生成一份学生入学登记表G_0，那么只要先用规范关系存贮各种介质的基本数据，如照片A_1(类型：Image)、简历A_2(类型：Text)、成绩A_3(类型：Table)、评语A_4(类型：Text)，然后定义G_0：

Define $G_0(A_1\{(1, 1), (8,10)\}, A_2\{(10, 1), (40, 10)\}, G_1(A_3\{(1,12),(40,18)\}, A_4\{(1, 20), (40, 22)\}))$

G_0对应的嵌套结构及显示布局如图4所示

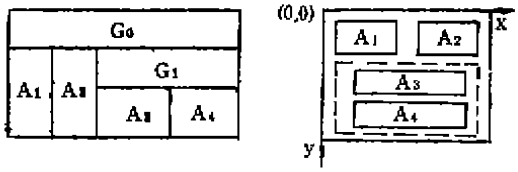

图4 G_0的嵌套结构及其显示布局

显然，图4中的嵌套结构等价于一棵树，为了与G_0的定义形式一致，我们按前序存放嵌套关系和属性数据字典：$G_0 \rightarrow A_1A_2G_1A_3A_4$，其具体描述形式如图5所示。

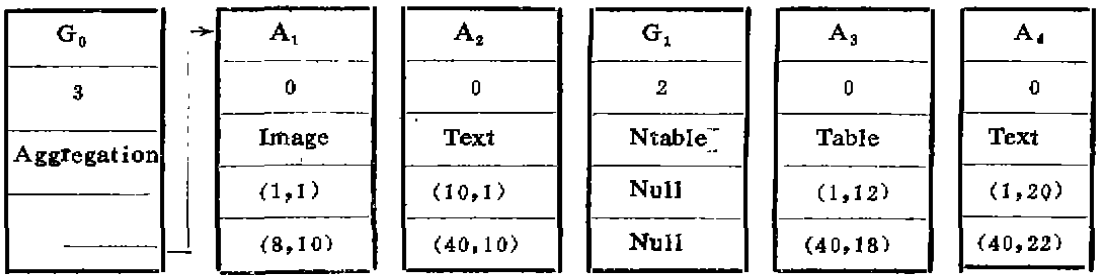

图5 G_0的嵌套结构及其布局

用NF^2方法对规范关系加以描述和构造具有如下优点：

1. 数据共享：包括基本数据（如A_1, A_2, A_3, A_4）和复杂的包括多种介质的数据（如G_0, G_1），在上例中如果用户已定义了G_1：

Define $G_1(A_3\{(1, 12), (40, 18)\}, A_4\{(1, 20), (40, 22)\})$

那么G_0的定义就变得更加简单：

Define $G_0(A_1\{(1, 1), (8, 10)\}, A_2\{(10, 1), (40, 10)\}, G_1)$

2. 数据管理简单一致：由于存贮描述非常方便，多介质数据的定义、操纵及生成均可以很容易的实现。

3. 支持抽象语义：根据规定的语法，可以自动捕捉抽象语义信息，如Generation，Aggregation等，清楚地表示了嵌套结构中层次之间的关系。

四、多介质数据的操纵

由于各种介质的基本数据存放完全独立于NF^2的构造描述，使得多介质数据的操纵不需要改变基本数据的物理存贮，而仅仅需要相应地更改NF^2的嵌套结构描述。因此，用户可以任意地组织所需的数据，如增加、删除、修改各种介质数据，改变多介质数据的显示布局等等，这种友好的、自由的多介质数据动态共享方式使系统具有强大的生命力。

为论述方便，假设G表示多介质数据，A表示各种介质的基本数据，则任一个多介质数据G均可表示为：

G:=A |cc |A, cc

A:=‹basic-media› |A, ‹basic-media›

cc:=G| cc, G

‹basic-media› :=digit |character |text |graph| image |voice

也就是说，G是由若干个基本数据嵌套生成，即$G=(G_1, G_2, \cdots, G_n)$。

如果记S_G为G本身及其所有子嵌套结构的集合，$S_G=\{G, G_1, G_2, \cdots, G_n, \cdots\}$，则不难给出多介质数据的操纵定义：

定义4.1 插入：设有多介质数据$G=(G_1, G_2, \cdots, G_n)$，又设$G_i \in S_G$，则任一个多介质数据$G_j \notin S_G$可以增加到G的$G_i$中生成新的多介质数据$G'$，其操作称为插入。显然：

$$G'=(G_1, G_2, \cdots, G_n)$$
$$\vdots$$
$$G_i=(G_{i1}, G_{i2}, \cdots, G_{ik}, G_j)$$
$$\vdots$$
$$G_j=(G_{j1}, G_{j2}, \cdots, G_{jk})$$
$$\vdots$$

例4.1 图4的多介质数据$G_0=(A_1, A_2, G_1)$，$S_G=\{G_0, A_1, A_2, G_1, A_3, A_4\}$，如果用户需要插入多介质数据$G_2=(A_5, A_6, A_7)$到$G_1$中，则生成的新的多介质数据$G_0'$：

$G_0'=(A_1, A_2, G_1)$

$G_1=(A_3, A_4, G_2)$

$G_2=(A_5, A_6, A_7)$

其对应的嵌套结构如图6所示：

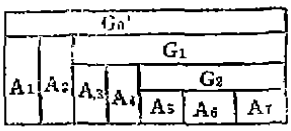

图6 G_0'的嵌套结构

插入多介质数据G_2时，可以执行以下语句：

Insert into G_1 inG:$G_2(A_5\{(x_{51}, y_{51}), (x_{52}, y_{52})\}, A_6\{x_{61}, y_{61}), (x_{62}, y_{62})\}, A_7\{(x_{71}, y_{71}), (x_{72}, y_{72})\})$

系统自动生成新的显示布局并修改嵌套关系和属性数据字典：

$G_0' \rightarrow A_1A_2G_1A_3A_4G_2A_5A_6A_7$

定义4.2 删除：设有多介质数据$G=(G_1, G_2, \cdots, G_n)$，则任一$G_i \in S_G$可以从G中去掉生成新的多介质数据$G'$，其操作称为删除。

例4.2 $G \rightarrow A_1A_2G_1A_3A_4 \xrightarrow{\text{删除}A_2} G' \rightarrow A_1G_1A_3A_4$

定义4.3 修改：设有多介质数据$G=(G_1, G_2, \cdots, G_n)$，则任一$G_i \in S_G$可以替换为多介质数据$G_j \notin S_G$，生成新的多介质数据$G'$，其操作称为修改。

例4.3 $G \rightarrow A_1A_2G_1A_3A_4 \xrightarrow[G_2=(A_5, A_6, A_7)]{\text{替换}G_1\text{为}} G' \rightarrow A_1A_2G_2A_6A_6A_7$

不难看出，通过插入、删除、修改操作及座标的变化，可以达到多介质数据的平移、旋转[8]等目的，从而用户能够根据需要随意改变多介质数据的显示布局。

五、结束语

我们用C语言在SUN工作站上实现了一个多介质数据库的实验系统，目前系统可以

很好地支持数字/字符、文字、图形等介质数据的存贮及构造。实践证明，NF^2 构造方法及其字典描述具有概念简单、处理方便的特点。今后准备建立一整套类似于SQL的多介质数据库管理语言以适应对各种介质类型的统一定义及操作。

参考文献

[1] Mark, A. R., Henery, F. K. etc, Extended Algebra and Calculus for Nested Relational Databaces, ACM Trans. on Databace Systems, 13:4 (1988), 389—417

[2] Levene, M. and Loizou, G., The Nested Relational Type Model, An Application of Domain Theory to Database, The Computer Journal, 33:1 (1990), 19—30

[3] Lawrence, A. R. and Michael, R. S., The POSTGRES Data Model, Proceedings of the 13th VLDB Conference, 1987

[4] Darrell, W. and Won, K., Multimedia Information Management in a Object-Oriented Database System, Proceedings of the 13th VLDB Conference, 1987

[5] John, P. and Fred, M., Semantic Data Models, ACM Computing Surveys, 20:3 (1988), 153—188

[6] 田沧海，林钧海，扩充三级模式支持多介质数据库，计算机研究与发展，27:12(1990)，12—16

[7] 田沧海，郭红英，林钧海，基于数据字典的用户友好接口，小型微型计算机系统，9:9 (1988),15—21

[8] Dadam, P. and Linnemann, V., Advanced Information Management (AIM): Advanced database technology for integrated application, IBM Systems Journal, 28:4(1989), 681—681

[9] Kemper, A. and Wallrath, M., An Analysis of Geometric Modeling in Database Systems, ACM Computing Surveys, 19:1(1987), 47—94

全国人工智能基本问题研讨会圆满成功

全国人工智能基本问题研讨会于1991年11月4日—7日在安徽省旌德县黄山旅游保健中心召开。会议由安徽省计算机学会和中国科技大学计算机系举办，得到中国人工智能学会、中国计算机学会和中国科技大学的大力支持。到会的有国内知名的教授级专家十九人，中青年讲师、博士生、硕士生十一人，共三十人。

会议首先由发起人、中国科技大学计算机系蔡庆生教授致开幕词，由安徽省计算机学会理事长、中科技大学计算机系主任钟津立教授致欢迎词，并由中国人工智能学会理事长涂序彦教授致词。会议收到了李家治、徐家福、何志均、刘叙华、王遇科、王树林、石纯一、林尧瑞、史忠植等教授的贺信或贺电。

会议在"人工智能当前该做些什么？"的中心议题下就人工智能的历史、现状与未来，包括：(1) 人工智能已取得的主要成果，(2) 人工智能目前存在的问题及原因分析，(3) 人工智能当前各学派之争，(4) 人工智能的发展方向与预期成果，(5) 中国人工智能未来十年的发展战略与策略，(6) 人工智能如何面向国民经济主战场等问题进行了专题报告、专题讨论与自由讨论。在会上作专题报告的共有十七位专家、教授。他们从数学、逻辑学、心理学、生理学、语言学、电子学、计算机科学、计算机工程、系统科学、思维科学等方面深入讨论了人工智能的若干基本问题。会议学术气氛十分浓厚，讨论非常热烈，真正作到了畅所欲言，相互促进，取长补短，求同存异。会议代表一致认为这次会议非常必要和及时，相当圆满和成功，具有深远意义。

会议代表对中国人工智能未来发展的战略与策略提出了许多好的思想与建议。会议认为，今后不定期地继续组织这种小型的、教授专家级的人工智能专题研讨会是很有必要的。

全国人工智能基本问题研讨会秘书组供稿