多维数据模型 $ER(\mathcal{H})$

李盛恩[1),2)] 王 珊[1)]

[1)](中国人民大学信息学院 北京 100872)

[2)](山东建筑工程学院计算机系 济南 250014)

摘 要 $ER(\mathcal{H})$把成员形式化为论域上的概念,成员之间的关系由概念的抽象程度所确定,层是成员的集合,层之间的关系由层中的成员所决定,成员之间的半序关系和层之间的半序关系统一用 OEM 图表示,采用路径表达式作为查询语言,从而把维抽象为一个半结构化对象,可以作为关系模型中的域. $ER(\mathcal{H})$给出了一组完整性规则. 扩充了关系模型和关系代数,使得能用关系对象模型表示多维数据. $ER(\mathcal{H})$可以方便地在主流的关系对象数据库上实现,并可以用于 XML 环境.

关键词 数据模型;数据仓库;联机分析;多维数据模型

中图法分类号 TP311

Multidimensional Data Model $ER(\mathcal{H})$

LI Sheng-En[1),2)] WANG Shan[2)]

[1)](*School of Information, Renmin University of China, Beijing* 100872)

[2)](*Department of Computer, Shandong Institute of Architecture & Engineering, Jinan* 250014)

Abstract $ER(\mathcal{H})$ defines members as abstract category on dimensional domain. The relations between members are determined by the abstract ability of category. Levels are collection of members. Relationships of levels are consistent to the relations of their members. $ER(\mathcal{H})$ uses Object Exchange Model to represent the partial order in members and levels, path expression as query language. Thus, dimension is considered as a semi-structure object and used as the domain of relation data model. $ER(\mathcal{H})$ also gives a set of integrity rules. $ER(\mathcal{H})$ expends relation model and relation algebra and uses object oriented and relation model to present multidimensional data. $ER(\mathcal{H})$ can be implemented easily on main stream relation-object data base systems and applied to XML environment.

Keywords data model; data warehouse; OLAP; multi-dimensional data model

1 引 言

联机分析是数据仓库中一种十分重要的应用,用于对数据进行多角度多层次的复杂分析. 为了达到这个目的,联机分析采用了多维数据模型. 直观地讲,多维数据模型是一个多维空间,维表示用户的观察对象,空间中的点表示度量的值. 例如,在分析产品的销售情况时,用户感兴趣的对象有时间、商品、地区和销售量,可以把时间、商品和地区作为维,销售量作为度量. 这样,空间中的点就表示某种商品在某个地区某时的销售量. 为了便于分析,每个维还可以被组织成若干个层,层之间的半序关系构成了一个或多个层次,如日、月、年三个层构成了时间维的

收稿日期:2003-06-07;修改稿收到日期:2005-11-01. 本课题得到国家自然科学基金(60273017,60496325)和国家"八六三"高技术研究发展计划项目基金(2002AA4Z3420)资助.李盛恩,男,1963 生,博士,教授,研究领域为数据仓库、联机分析和 XML 数据管理. E-mail: lsn@sdai.edu.cn. 王 珊,女,1944 年生,教授,博士生导师,主要研究领域为数据库系统和知识工程、数据仓库.

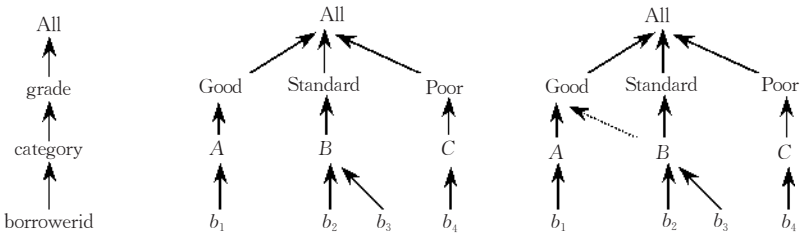
一个层次，日、周、月、年构成了另外一个层次. 联机分析常用的操作包括沿着一个维的某个层次的上钻和下钻(roll up 和 drill down)操作，选择立方体中部分数据的切块和切片(dice 和 slice)操作以及为了展现的旋转操作(pivot).

多维数据模型是随着联机分析产品的流行而出现的，缺乏坚实的理论基础. 理论界从不同的方面提出了一些模型[1~9]，微软也提出了 OLE DB for OLAP①. 但是还存在以下的问题：

(1)成员之间的关系从属于层之间的关系

在多维数据模型中，成员之间和层之间都存在半序关系. 目前的多维数据模型采用先定义层之间的半序关系，然后在两个相邻层定义一个 *R-UP* 函数描述这两个层中的成员之间的关系，非相邻层成员中的关系用 *R-UP* 函数的复合函数表示. 无法描述实际应用中的一些复杂问题.

例 1. 图 1 给出了信用等级维的一个层次和两个实例. 对实例 1，在层 borrowid 和 category 之间定义函数 $R\text{-}UP_1$，把借款人 b_1 映射到类别 A，b_2 和 b_3 映射到类别 B，b_4 映射到类别 C. 在层 category 和 grade 之间定义函数 $R\text{-}UP_2$，把类别 A，B，C 分别映射到等级 Good、Standard 和 Poor. 借款人和等级之间的关系则由复合函数 $R\text{-}UP_2 \circ R\text{-}UP_1$ 决定. 如果由于某种商业规则，借款人 b_3 虽然属于类别 B，但必须属于等级 Good，如实例 2 所示. 然而，利用复合函数机制 b_3 将被映射到 Standard，导致错误的结果.



(a) 层次 (b) 实例 1 (c) 实例 2

图 1 信用等级维层次以及实例[10]

(2)同一层的成员有相同的属性

成员除了名字外还有很多属性，可用于特征分析，多数模型要求同一层中的成员具有相同的属性. 在实际应用中，会遇到即使是同一层的成员也会有不同的属性[5]. 例如，电视机和洗衣机都属于产品维的同一个层，但是洗衣机有洗衣重量属性，而电视机却没有该属性，电视机有制式属性，而洗衣机没有.

(3)没有显式的完整性限制

为了加快查询响应时间，联机分析产品大量地采用了基于实例化视图的查询重写技术，多维数据模型隐含的要求 *R-UP* 函数是一个单值的、完全的、满射函数以方便查询重写. 但是在实际应用中有无法定义满足条件的函数的情况，这时需要一个显式地说明完整性限制的方法.

例 2. 假设地区维有一个层次：store→city→province→country→All. 可以在 store 和 city 之间定义一个满足要求的 *R-UP* 函数，把 store 中的每个成员映射到 city 中的一个成员，并保证 city 中的任何一个成员至少是 store 中一个成员的象. 但在层 city 和 province 之间，由于像北京、上海等一些直辖市的存在，无法定义满足要求的函数(不能把 city 中的所有成员映射到 province 中的某个成员). 因为目前的模型缺乏一种机制来说明这种异常情况，如果使用在层 province 上定义的实例化视图重写对层 country 的查询，将导致错误的结果.

(4)缺乏对度量和聚集函数的形式化定义

联机分析中会遇到各种各样的度量和复杂的聚集函数，目前的模型没有把度量和聚集函数作为模型的组成部分，使用不当会导致错误的结果.

例 3. 假设我们知道某系在过去三年中的在校研究生数分别是 15，17 和 13，把这三个数的和 45 作为该系三年来的招生数是错误的，因为在校研究生数是 stock 类型的度量，该类型的度量不能沿着时间维做聚集运算[11]. 如果我们知道了某种产品每月的日平均销售量，不能从每月的日平均销售量得到全年的日平均销售量，因为日平均销售量是平均函数的结果，而平均函数不具有分布特性.

为了克服现有模型的缺陷，我们提出了多维数据模型 $ER(\mathcal{H})$，其主要思想是用关系对象模型表达多维数据模型. 与其它的模型相比，我们的模型有以下的特点：

(1)成员是一个概念，维是一个代数系统.

我们把成员形式化定义为一个概念，成员之间的关系由概念的抽象程度所决定. 层是成员集合，表

① http://msdn.microsoft.com/library/en-us/oledb/htm/olap-salesdata_example_dataset_expression.asp

示一种分类方法，层之间的关系由层中的成员所决定. 维是一个代数系统$\langle M,O\rangle$，M 是成员集合，O 是 M 上的基本运算. 基本运算有判定两个成员的包容关系、求出属于一个层的所有成员、给出具有某个属性值的成员、一个成员的后代或祖先等等.

(2)给出了三个完整性限制，防止错误的查询重写.

(3)扩充了度量维的概念，引入了度量属性和度量表达式，防止误操作.

例如$average_D(\pi_{A,\mathrm{year},C;D}(average_D(\pi_{A,\mathrm{month},C;D}(c))))$是一个符合语法规则的查询语句，内层的 average 使得度量 D 的度量表达式为 $average(c.D)$，当执行外层的 average 时，由于是对一个 average 的结果再次进行 average 运算，系统将拒绝执行，保证不输出错误的结果.

(4)聚集函数作为模型的一部分，具有统一的说明方式.

由于 OLAP 需要种类繁多的函数，特别的，应该允许用户自定义函数，我们采用文献[12]的结果，统一了函数的说明方式，便于优化和施加完整性限制.

本文第 2 节给出了成员、层、维和 Cube 的形式化定义；第 3 节提出了一组完整性规则；第 4 节扩充了选择、投影和连接三个关系操作符，给出了一些应用实例；第 5 节讨论了相关工作；第 6 节总结了全文.

2　形式化定义

在这一节中，我们给出模型的形式化定义. 用$\mathcal{N}$表示名字的集合，$\mathcal{A}$表示属性的集合，$\mathcal{U}$表示论域，即感兴趣的对象组成的集合.

定义 1.　成员模式是一个三元组(M,A,S)，$M\subseteq\mathcal{N}$，$A\subseteq\mathcal{A}$，$S\subseteq 2^{\mathcal{U}}$.

定义 2.　成员实例(简称成员)是一个变长元组$(m,\{A_1:a_1,A_2:a_2,\cdots,A_n:a_n\},s)$，其中 $m\in M$，$A_i\in A$，$a_i\in Dom(A_i)$，$Dom(A_i)$是属性$\mathcal{A}_i$的域，$1\leq i\leq n$，$s\in S$. $(\mathrm{All}_m,\{\},\mathcal{U})$是一个特殊成员，简记为 All_m. 基本成员为$(m,\{A_1:a_1,A_2:a_2,\cdots,A_n:a_n\},\{e\})$，$e\in\mathcal{U}$，基本成员与$\mathcal{U}$的元素存在一一对应关系.

例 4.　假设$\mathcal{U}$是产品的集合，$\mathcal{U}=\{p_1,p_2,\cdots,p_n\}$，$\mathcal{A}$是属性的集合，$\mathcal{A}=\{$品牌，颜色，洗衣重量，制式$\}$. 表 1 给出了一些成员，为方便起见，表中只给出了属性值，没有给出对应的属性名. 前 2 个元组是基本成员，它们是不同品牌的电视机，第 3 个元组描述了成员电视机，它是一个抽象概念，包容了前 2 个元组. 最后一个元组描述成员 All_m，它代表所有的产品.

表 1　成员模式和成员实例

M	A	S
P_1	{Sony，黑，NTSC}	$\{p_1\}$
P_2	{Haier，白，PAL}	$\{p_2\}$
TV	{}	$\{p_1,p_2\}$
…		
All_m	{}	$\{p_1,p_2,\cdots,p_n\}$

定义 3.　层模式是一个二元组(L,S)，$L\subseteq\mathcal{N}$，$S\subseteq 2^M$，M 是$\mathcal{U}$上的成员集合.

定义 4.　层实例(简称层)是一个变长元组(l,s)，$l\in L$，$s\in S$. $(\mathrm{All}_l,\{\mathrm{All}_m\})$是一个特殊的层，简记为 All_l.

例 5.　假设$\mathcal{U}$是商店的集合，$\mathcal{U}$上的成员有 s_1，s_2，…，s_n，烟台，青岛，济南，…，山东，…，All_m. $\mathcal{U}$上的层模式和层实例如表 2.

表 2　层模式和层实例

L	S
All_l	$\{\mathrm{All}_m\}$
省	{山东，…}
城市	{烟台，青岛，济南，…}
商店	$\{s_1,s_2,\cdots,s_n\}$

在后面的叙述中，我们用 M 表示$\mathcal{U}$上的成员集合，m 表示成员，$m.A$ 表示成员 m 的属性和属性值对的集合，$m.S$ 表示成员 m 的定义域，$m.S\subseteq\mathcal{U}$. 用 L 表示层集合，l 表示层，$l.S$ 表示层 l 中的成员集合，用 $l.U$ 表示层 l 的定义域，$l.U=\{m.S\mid m\in l.S\}$.

成员之间和层之间存在着半序关系，我们使用成员关系图和层关系图来分别描述这两种关系.

定义 5.　成员关系图 $G_m(M,E_m)$是一个有向图，如果 $m_1\in M$，$m_2\in M$，$m_1\neq m_2$，$m_1.S\subseteq m_2.S$，并且不存在 $m_3\in M$，使得 $m_1.S\subseteq m_3.S\subseteq m_2.S$，则有 $(m_1,m_2)\in E_m$.

定义 6.　层关系图 $G_l(L,E_l)$是一个有向图，如果 $l_1\in L$，$l_2\in L$，并且 $\exists\, m_1\in l_1.S$，$m_2\in l_2.S$，$m_1.S\subseteq m_2.S$，而且不存在 $l_3\in L$，$m_3\in l_3.S$，使得 $m_1.S\subseteq m_3.S\subseteq m_2.S$，则$(l_1,l_2)\in L$.

例 6.　例 1 的实例 2 中的成员关系图和层关系图如图 2.

从图 2 可以看出，每个成员都得到了正确的定义，原因在于我们把成员看成是$\mathcal{U}$上的一个概念，每个成员都是在$\mathcal{U}$上定义的，没有使用复合函数. 两个层相邻只表示它们之间存在一个部分映射，例如，

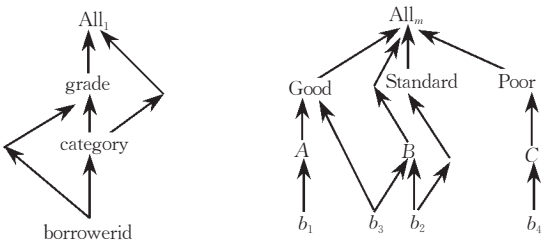


图 2 成员和层关系图

category→grade，但成员 B 没有被映射到 grade. 这样，查询重写就不能简单地根据层之间的相邻关系而决定，而必须引入某种显式的说明机制，以防止错误的结果.

为了能统一处理成员之间的半序关系和层之间的半序关系，我们引入 OEM 图[13] G. 将成员关系图和层关系图转换成 OEM 图的规则是：G_m 中的结点作为 G 中的结点，G_m 中的边作为 G 中的边，标签为 Child；对 G_l 中的任一边 (l_1, l_2)，如果成员 $m_1 \in l_1$，$m_2 \in l_2$，$m_1.S \subseteq m_2.S$，则在 G 的两个结点 m_1 和 m_2 之间增加一条边，标签为 l_1；如果成员 m_1 具有属性 A_1，属性值 a_1，则在 G 中增加结点 a_1，并在结点 m_1 和 a_1 之间增加一条边，标签为 A_1.

例 7. 例 4 中商店维的 OEM 图如图 3，图中省略了所有的 Child 标签. 成员山东有经理属性，其经理是张明. 成员 S_{im} 有属性库存，库存量为 960. 层城市的成员有烟台、青岛、济南和北京.

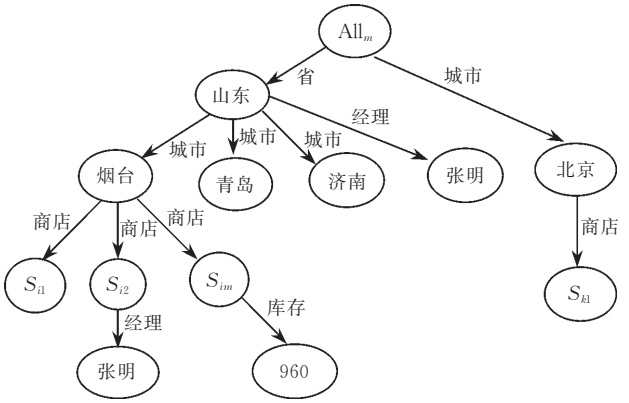


图 3 商店维的部分 OEM 图

定义 7. 维模式是一个二元组 $\langle D, G\rangle$，$D \subseteq \mathcal{N}$，G 是 OEM 图集合.

定义 8. 维实例(简称维)是一个名字为 d 的 OEM 图.

将维表示成 OEM 图以后，我们采用路径表达式作为对维的操作，路径表达式的详细介绍请参见文献[13].

定义 9. 简单路径表达式是一个序列，$Z.l_1 \cdots l_n$，其中 $l_1, l_2, \cdots, l_n$ 是 OEM 图中的标签，Z 是 OEM 中的结点或结点变量. 简单路径表达式还有另外一种书写形式 $Z.l_1 Z_1, Z_1.l_2 Z_2, \cdots, Z_{n-1}.l_n Z_n, Z_i$ 是结点变量. 简单路径表达式的值是数据路径，即 $o_0, l_1, o_1, \cdots, l_n, o_n.o_i$ 是 OEM 中的结点，对任何的 i，在 o_{i-1} 和 o_i 之间有标签为 l_i 的边. 对一条简单路径表达式会有多条数据路径满足该表达式.

例 8. 图 3 中满足简单路径表达式 All_m.省.城市的数据路径有：All_m，省，山东，城市，烟台；All_m，省，山东，城市，青岛；All_m，省，山东，城市，济南.

为了使用简单路径表达式表示维上的操作，我们把简单路径表达式扩充为下面的形式：

$$Z.l_1 Z_1, Z_1.l_2 Z_2, \cdots, Z_{n-1}.l_n Z_n; Z_i,\ 1 \le i \le n.$$

简单路径表达式的值是变量 Z_i 的值，$Z_i = \{o_{i1}, o_{i2}, \cdots, o_{ik}\}$，$o_{ij}, 1 \le j \le k$ 是某条满足简单路径表达式的数据路径中的第 i 个对象.

例如，有一条路径表达式 All_m.省 Z_1, Z_1.城市 $Z_2; Z_2$，该表达式的值为{烟台，青岛，济南}. 表达式 All_m.省 Z_1, Z_1.城市 $Z_2; Z_1$ 的值为{山东}. 从中去掉我们不关注的变量，这两个表达式可以简写为 All_m.省.城市 $Z_2; Z_2$ 和 All_m.省 Z_1, Z_1.城市；Z_1.

定义 10. 通用路径表达式是简单路径表达式的扩展，即 Z 后面可以跟一个或多个 gpe-component，而不只是一个标签串. gpe-component 的语法如下：

1. 如果 l 是一个标签，则.l 是一个 gpe-component；

2. 如果 X 是一个对象，则.*unquote*(X)是一个 gpe-component；

3. 如果 s_1 和 s_2 是 gpe-component，则下面也都是 gpe-component.

$$s_1 s_2, s_1 | s_2, (s_1), (s_1)?, (s_1)+, (s_1)*$$

其中，|表示或，指前面的 gpe-component 出现 0 或 1 次，+指前面的 gpe-component 出现 1 或多次，*指前面的 gpe-component 出现 0 或多次. 函数 unquote 返回变量的值.

在 gpe-component 允许出现通配符%，%匹配标签中的 0 或多个任意字符. 通配符.# 表示长度大于等于 0 的数据路径. .# 实际上是(.%)* 的简写形式.

例 9. 表达式 All_m.#.城市 $Z; Z$，给出了层城市中所有的成员. 表达式 All_m.# Z_1, Z_1.经理 Z_2, Z_2 = 张明；Z_1，返回经理是张明的所有成员.

在联机分析中会有很多的度量需要分析，例如，销售量、销售收入、销售成本和利润等. 在生命科学等领域，有成百上千的实验项目，每个项目都是一个

度量[14]. 这样不利于查询和展示. 微软提出了度量维①的概念，它是所有度量的集合. 我们采用微软的基本思想，并引入度量属性描述度量的特性和度量之间的关系.

度量的特性可以有 flow，stock，value-per-unit，atom 和 derived. atom 表示度量是来自源数据，derived表示需要从其它的 atom 度量经过计算才能得到值. 引用文献[11]的定义，flow 表示一个时间段并在时间段的终点记录的值，stock 是测量并记录在某个时间点上的值，value-per-unit 也是某个时间点的值，但是与计量单位有关. 例如，销售量具有 atom 和 flow 属性，利润具有 derived 和 flow 属性，因为，一般地讲利润是需要经过计算的. 库存量是典型的具有 stock 属性的量. 某种商品的单价具有 value-per-unit 属性.

定义 11. 度量维模式为 (M,T,P,E)，$M\subseteq\mathcal{N}$，$Dom(T)=\{\text{Integer},\text{Real},\cdots\}$，$Dom(P)=\{\text{flow},\text{stock},\text{value-per-unit},\text{atom},\text{derived}\}$，$E$ 是算术表达式，其中的一个特殊表达式为 Null，表示一个空白表达式.

定义 12. 度量实例(简称度量)是度量维模式上的一个元组.

例 10. 产品销售涉及到的度量有销售量、单价和销售金额，把它们组织成一个度量维，度量实例如表 3 所示.

表 3　度量表

M	T	P	E
销售量	Integer	{atom，flow}	Null
单价	Real	{atom，value-per-unit}	Null
销售金额	Real	{derived，flow}	销售量×单价

聚集函数的作用是对一组度量值进行计算得到一个值. OLAP 应用需要很多统计函数，特别的，应该允许用户定义自己的函数. 聚集函数使用不当，会导致错误的结果. 因此有必要把聚集函数作为多维模型的一部分. 文献[12]详细讨论了 UDA(User Defined Aggregates)，并给出了一个高层的定义语言 SADLE，每一个聚集函数被抽象为四部分：State、Initialize、Iterate 和 Terminate. State 存储中间计算结果，Initialize 完成初始化工作，Iterate 反复处理一组数值，Terminate 决定函数的终止时机.

受文献[12]启发，我们给出下面的聚集函数定义.

定义 13. 一个聚集函数是一个八元组$\langle Name, Property, Input, Output, State, Initialize, Iterate, Terminate\rangle$，$Name$ 表示聚集函数的名字，$Input$ 和 $Output$ 是函数的输入和输出. $Property$ 表示函数的特性，$Dom(Property)=\{distribute, algebraic, holistic\}$[15]. $Initialize$，$Iterate$ 和 $Terminate$ 是过程名字.

定义 14. Cube 模式是一个扩展的关系模式 $R(A_1,A_2,\cdots,A_m)$，对任何的 $A_i(1\le i\le m-1)$，$Dom(A_i)=d.M$，d 是一个维，$d.M$ 是 d 的成员集合，A_m 是度量维，$Dom(A_m)=Dom(m_k)\times Dom(m_{k+1})\times\cdots\times Dom(m_l)$，$m_j\in M$，$k\le j\le l$.

定义 15. Cube 实例(简称 Cube)是一个扩展的关系 r，对任何一个元组 $t\in r$，有 $t\in Dom(A_1)\times Dom(A_2)\times\cdots\times Dom(A_m)$.

Base Cube 是一个 Cube，在它的模式中包含了所有的维和度量维中所有具有 atom 属性的度量，并且每个维的取值都是基本成员.

对每个维 d，我们定义一个同名的 Cube 模式 $R(N)$，其实例中是 d 中的所有成员集合.

例 11. 假设有时间维 T、产品维 P、商店维 S 和度量维 M. 在这些维上定义了一个 Cube 模式，$R(A_1:T,A_2:P,A_3:S,A_4:M\{\text{销售量}\})$. $A_1:T$ 表示属性 A_1 的域是时间维 T，$A_4:M.\{\text{销售量}\}$表示属性 A_4 的域是度量维中的度量销售量的域. 该模式的一个实例如表 4(a)所示，表 4(b)是一个 Base Cube，它包含了度量维中所有具有 atom 属性的度量，即销售量和单价.

表 4　Cube 实例

(a) Cube

A_1	A_2	A_3	A_4
2003/4/1	P_1	S_{i1}	10
2003/4/1	P_2	S_{j1}	30
2003/4/1	P_3	烟台	100
2003/4/1	P_4	S_{k1}	11
2003/4/1	TV	山东	450

(b) Base Cube

A_1	A_2	A_3	A_4
2003/4/1	P_1	S_{i1}	(10,1800)
2003/4/1	P_1	S_{j1}	(13,1800)
2003/4/1	P_2	S_{k1}	(46,3600)
2003/4/1	P_2	S_{j3}	(69,3400)

3　完整性规则

在 $ER(\mathcal{H})$模型中，因为每个成员都是在论域 $\mathcal{U}$

① http://msdn.microsoft.com/library/en-us/oledb/htm/olap-salesdata_example_dataset_expression.asp

上定义的,或者说是在基本成员上定义的(因为基本成员与 U 中的元素存在一一对应关系),所以在原始数据中必须出现所有的基本成员,否则,某些成员的统计结果就不正确,而原始数据存放在 Base Cube 中. 层是成员的集合,是为了便于对原始数据进行综合分析而提出的一种分类方法,是基本成员集合或其子集的一个划分. 因此,在 $ER(\mathcal{H})$ 中存在两种完整性约束:

(1)基本成员完整性. 对 Base Cube C 的任何一个维属性 A,$Dom(A)=d.M$,有 $U_d=\pi_A(C)$,其中,U_d 是维 d 的基本成员集合.

(2)层完整性. 对任意的层 l,其成员必须是两两不相交的,即对任意的两个成员 $m_1\in l$,$m_2\in l$,有 $m_1.S\bigcap m_2.S=\Phi$.

在其它的多维数据模型中,如文献[7],往往要求成员层次关系图是一颗树,两个相邻层之间存在着单值、完全、满射函数,这样的限制过于严厉,无法表示例 1 中的实例 2. 除掉这些限制后,就不能简单地根据 $l_1\rightarrow l_2$ 而用 l_1 重写对 l_2 的查询. 需要根据成员和层的定义以及聚集函数的特性决定.

查询重写完整性:对查询 $Q(l_2,f)$,其中,f 是聚集函数,l_2 是层,必须满足以下的条件,才可以用 l_1 重写查询 Q:

(1) f 是分布函数,即 $f.Property=distribute$;

(2) $l_1.U=l_2.U$;

(3) $\forall m_1\in l_1$,$\exists m_2\in l_2$,$m_1.S\subseteq m_2.S$.

4 扩展的关系代数

因为 Cube 以维作为域,因此需要对关系操作进行扩展以表达维的结构等语义信息. 在这一节,我们扩展了关系代数的选择、投影和连接操作,并提出了一个新的聚集操作.

选择操作

关系模型的选择操作形式为:$\sigma_F(r)=\{t\mid t\in r\wedge F(t)=\text{true}\}$. 对选择操作的扩充主要体现在选择条件中. 除了关系模型中允许的表达式以外,在 F 中还可以出现以下形式的表达式:

$$A\{=,\subseteq\}Z.gpe\text{-}component;Z_i$$

其中,A 是 r 的属性 $Z.gpe\text{-}component;Z_i$ 是 A 的域(维)上的路径表达式. 运算符 $=$ 和 $\subseteq$ 是集合的相等运算和包含运算.

投影操作

关系模型的投影操作形式为 $\pi_A(r)=\{t[A]\mid t\in r,A\in DS\}$,$DS$ 是 r 的模式. 关系模型的投影运算要去除重复元组(因为关系是集合),在 SQL 中,可以选择是否去除重复元组. 在多维数据模型中,投影操作的目的主要是为了做聚集运算. 文献[6]提出了 Packing 操作,用于将在某些属性上具有相同值的元组合并到一起. 我们把投影操作形式化为具有 Packing 能力的操作符.

$$\begin{aligned}\pi_{A;B}(c)=\{t[AB]\mid\exists t_1,t_2,\cdots,t_m\in c,\\ t_1(A)=t_2(A)=\cdots=t_m(A)=t(A),\\ t(B)=t_1(B)\bigcup\cdots\bigcup t_m(B)\},\end{aligned}$$

其中,A 和 B 是 c 模式中的属性集,A 叫做分类部分,B 叫做聚集部分,B 的域可以是维也可以是度量维. $\bigcup$ 取包(bag)语义. 如果 A 为空集,则将 c 中的所有元组投影为一个元租. 如果省略 B,上面的操作符就是一般的关系投影运算.

连接操作

关系模型的连接操作为 $r\underset{\theta}{\bowtie}s=\{t(RS)\mid\exists t_r\in r,t_s\in s,t(R)=t_r(R),t(S)=t_s(S),t_r\theta t_s=\text{true}\}$. 除了关系模型中允许的表达式以外,在 θ 中还可以出现以下形式的表达式:

$$r.A.gpe\text{-}component;Z\{=,\subseteq\}s.B \text{ 或}$$

$$r.A\{=,\subseteq\}s.B.gpe\text{-}component;Z$$

聚集操作

用 f 表示聚集函数,我们将聚集操作定义为

$$\begin{aligned}f_A(c)=\{t\mid\exists t_1\in c,t(R-A)=t_1(R-A),\\ t(A)=f(t_1(A))\}\end{aligned}$$

其中,R 是 Cube C 模式中的所有属性,A 是某一个属性,A 的域可以是维也可以是度量维.

下面我们通过一些例子来说明 $ER(\mathcal{H})$ 的表达能力.

例 12. Base Cube C 的模式是 $R(A:$商店维,$B:$度量维{销售量}),用 S 表示商店维,商店维的一个实例如图 3 所示.

Q_1:给出商店 S_{i1} 的销售量.

$\sigma_F(R)$,其中 $F:A=S_{i1}$

Q_2:给出所有经理名字叫张明的商店的销售量.

$\sigma_F(R)$,其中 $F:A\subseteq Z_1$. 商店 Z_2,Z_2. 经理 Z_3,$Z_3=$张明;Z_2

Q_3:给出库存量大于 100 万,销售量小于 10 万的商店的销售量.

$\sigma_F(R)$,其中 $F:A\subseteq Z_1$. 商店 Z_2,Z_2. 库存 Z_3,$Z_3>100;Z_2$ and $B<10$

Q_4:给出每个城市的销售量.

$R_1=sum_{R,B}(\pi_{S;R.B}(\sigma_F(S)\underset{\theta}{\bowtie}R))$,其中,

$F:S\subseteq \mathrm{All}_m.\#.$城市 $Z;Z$

$\theta:R.A\subseteq S.$商店 $Z;Z$

Q_5：给出每个省各个城市的销售量.

$R_2=\sigma_F(S)\underset{\theta}{\bowtie}R_1))$，其中，

$F:S\subseteq \mathrm{All}_m.$省 $Z;Z$

$\theta:R_1.S\subseteq S.$城市 $Z;Z$

Q_6：给出每个直辖市的销售量.

$R_3=sum_{R.B}(\pi_{S;R.B}(\sigma_F(S)\underset{\theta}{\bowtie}R))$，其中，

$F:S\subseteq \mathrm{All}_m.$城市 $Z;Z$

$\theta:R.A\subseteq S.\#.$商店 $Z;Z$

Q_7：给出全国的销售量.

$R_4=R_3\bigcup sum_{R.B}(\pi_{S;R.B}(R_2))$

$sum_{R.B}(\pi_{S;R.B}(R_4))$

Q_1与一般的关系选择语句一样，因为 Cube 本身就是一个扩展的关系. Q_2和 Q_3查询具有某个属性值的成员. Q_4和 Q_5是上钻和下钻操作. Q_6使用了直辖市是直接隶属于国家的知识，其它的模型不能表达该查询. Q_7利用了全国的销售量是各省、直辖市的销售量之和的知识.

例 13.　文献[7]给出了一个多维数据库和一组查询. 多维数据库有 location，time 和 product 维，sales 为度量. location 维有唯一的维层次：store→city→state→region→country→All，每个层有属性 manager. Base Cube 的模式是 $R(A:\mathrm{time},B:\mathrm{location},C:\mathrm{product},D:\mathrm{sales})$，用 T,L 和 P 分别表示 time，location 和 product 维.

Q_1：找出由 john smith 管理的所有 location.

$R_1=\sigma_F(L)$，其中，$F:L\subseteq Z.\mathrm{manager}\ Z_1$，

$Z_1=\mathrm{john\ smith};Z$

Q_2：找出 USA 在 NE 地区的每个 location（在任何一层）的总销售量.

$R_2=sum_{R.D}(\pi_{L;R.D}(\sigma_F(L)\underset{\theta}{\bowtie}R))$，其中，

$F:L\subseteq \mathrm{USA.region.NE}\ Z_1,Z_1(.\%)+Z_2;Z_2$

$\theta:R.B\subseteq L$

Q_3：找出销售量超过 100000 的所有 location（在任何一层）.

$R_3=\sigma_F(sum_{R.D}(\pi_{L;R.D}(L\underset{\theta}{\bowtie}R)))$，其中，

$\theta:R.B\subseteq L$

$F:R.D>100000$

Q_4：对每个销售量超过 100000 的 location，给出与它们直接相邻的下一层的销售量.

$R_4=sum_{R.D}(\pi_{R3.L,L;R.D}(R_3\underset{\theta_1}{\bowtie}L\underset{\theta_2}{\bowtie}R))$，其中，

$\theta_1:L\subseteq R_3.L.\mathrm{Child}\ Z;Z$

$\theta_2:R.B\subseteq L$

Q_5：找出销售量超过 100000 的所有 location 和销售量小于它的 10%的直接相邻的下一层.

$R_5=\pi_{R_3.1,R_4.2}(R_3\underset{\theta}{\bowtie}R_4)$，其中，

$\theta:R_3.1=R_4.1\ \mathrm{and}\ R_4.D<10\%\times R_3.D$

Q_6：找出在销售量超过 100000 的 location 的直接或间接的下层中销售状况不好的 product（销售量小于 1000）.

$R_6=\sigma_F(sum_{R.D}(\pi_{L,R.C;R.D}(R_3\underset{\theta_1}{\bowtie}L\underset{\theta_2}{\bowtie}R))$，其中，

$\theta_1:L\subseteq R_3.1(.\%)+Z;Z$

$\theta_2:R.B\subseteq L$

$F:R.D<1000$

例 14.　微软[①]有一个查询销售员 Venkatrao 和 Netz 在美国北部各州、美国南部、日本在 1991 年的第一季度各月、第二和三季度、第四季度各月所有产品的销售业绩的应用. 多维数据库有 time，location，product 和 person 维，度量维中有一个度量 sales. time 有层次 day→month→quarter→year→All，location 有层次 city→state→region→country→All，其它的维也有层次，但与本查询无关. Base Cube C 的模式是 $R(A:\mathrm{time},B:\mathrm{location},C:\mathrm{product},D:\mathrm{person},E:\mathrm{sales})$，分别用 T,L,P 和 PS 表示 time，location，product 和 person 维.

为了清晰起见，我们分 3 步给出了查询的 Cube 代数：

$R_1=sum_E(\pi_{A,B,D;E}(\sigma_{(D=\mathrm{Venkatrao\ or\ }D=\mathrm{Netz})\ \mathrm{and}\ A.\mathrm{year}=1991}(R)))$

$R_2=\sigma_{L=\mathrm{Japan\ or\ }L\subseteq\mathrm{USA.region}\ Z_1,Z_1=\mathrm{north},Z_1.\mathrm{state}\ Z_2;Z_2\ \mathrm{or}\ L\subseteq\mathrm{USA.region}\ A_1,A_1=\mathrm{Sourth};A_1}(L)$

$R_3=\sigma_{T\subseteq 1991.\mathrm{Quarter}\ Z_1,Z_1=(\mathrm{Quarter1\ or\ Quarter4}),Z_1.\mathrm{month}\ Z_2;Z_2\ \mathrm{or}\ T\subseteq 1991.\mathrm{Quarter}\ A,A=(\mathrm{Quarter1\ or\ Quarter3});A}(T)$

$sum_E(\pi_{R_3.T,R_2.L,R_1.D;R_1.E}(R_3\underset{\theta_1}{\bowtie}(R_2\underset{\theta_2}{\bowtie}R_1)))$

其中，$\theta_2:R_1.B\subseteq R_2.L$

$\theta_1:R_1.A\subseteq R_3.T$

R_1从 R 中选择出 Venkatrao 和 Netz 于 1991 年在各个地方的销售量，这里已经去掉了 product 维. R_2从 location 中选择日本、美国北部各州和美国南部. R_3从 T 中选择 1991 年第一、四季度的各月、第二和三季度.

当采用 star-schema 实现多维数据模型时，表达这个查询的 MDX 语句需要分解为 6 个 star-join. 文献[16]利用多查询优化技术对这些 star-join 进行

① http://msdn.microsoft.com/library/en-us/oledb/htm/olap-salesdata_example_dataset_expression.asp

了优化.从上面可以看出,用 Cube 代数表达的查询十分清晰,是一个标准的 SPJ 语句.利用传统的关系优化技术即可以得到和文献[16]相同的查询执行计划.

5 相关工作

多维数据模型的研究工作基本可以分为两个方向:用关系模型表示多维数据模型[1,3,7]和把多维数据模型形式化为多维空间[2].

我们的工作主要与文献[7]相关.文献[7]提出了多维数据模型 $SQL(\mathcal{H})$,其主要思想是把维抽象为一个层次域$\langle U,O\rangle$,U 是成员的集合,O 是成员之间的比较运算集合,包括=、<(父子关系)、<=、<<(<的传递闭包)和<<=.并且要求由成员之间的父子关系导出的图 $G_<(U,E)$是一棵树,其中 $E=\{(m_1,m_2)\mid m_1\in U,m_2\in U,m_1<m_2\}$.这样的要求过于严厉,无法表示例 1 中的实例 2.另外,操作符比较简单,只能比较两个成员之间的父子或祖先关系,不能表示一些更为复杂的关系,例如,求出某个成员的第 3 代后继成员.

$SQL(\mathcal{H})$允许同一层的成员具有不同的属性,具体的做法是把同一层中有相同属性的成员存放在一个关系中,这样,同一层的成员分散在多个关系中.$SQL(\mathcal{H})$允许一个层中的成员可以映射到多个层中的成员,但是要求同一个关系中的所有成员必须映射到另外一个层的同一个关系中的某些成员.这样比较好地解决了引言中提出的问题 1 和问题 2,但是弱化了层的概念,以至于在文中的例子中没有出现过对层的操作,例如,求出某个层中的所有成员.一个重要的原因在于 $SQL(\mathcal{H})$仍然属于关系模型范畴.

与 $SQL(\mathcal{H})$相比,$ER(\mathcal{H})$明确将成员形式化定义为论域上的概念,成员之间的父子关系是概念之间的包容关系,层之间的半序关系由其中的成员之间的关系所决定,用一个 OEM 图同时表示成员之间的半序关系和层之间的半序关系,用路径表达式表示允许的操作,从而把维抽象为一个半结构化对象,查询表达能力比 $SQL(\mathcal{H})$更强.例如,可以方便地用路径表达式表示"求出某个成员的第 3 代后继成员"和"求出某个层中的所有成员".路径表达式的一个很大的优点是可以表示结构化方面的信息,例如,例 12 中的查询 Q_6:求出每个直辖市的销售量.因此,把维抽象为一个半结构化对象是本文的一个主要的创新点.

$ER(\mathcal{H})$去掉了其它模型中过于苛刻的限制,提出了三个完整性规则:层完整性反映了我们认为层是对基本成员的划分的认识,符合实际情况.基本成员完整性和查询重写完整性是为了使模型能用于处理复杂应用而提出的,其思想主要来自于文献[11]对统计数据库 Summarizability 问题的研究结果.文献[9]首先在多维数据模型中引入了完整性规则,完整性规则用类路径表达式表示,但其主要目的是为了解决查询重写的问题.文献[9]的工作可以用于扩充 $ER(\mathcal{H})$模型,将其提出的完整性规则作为用户自定义的完整性规则.

$ER(\mathcal{H})$将路径表达式的结果定义为 OEM 图中结点的集合,即成员的集合,以便将其用于关系的集合运算,并在其基础上扩展关系运算中的选择、投影和连接运算,提出了聚集操作运算.其中投影运算和聚集操作运算参考了文献[6]的 Packing 操作.

最后,需要说明的是,文献[10]针对例 1 中的实例 2 提出了解决办法,它仍然使用 *R-UP* 函数定义成员和成员之间的关系,把像实例 2 中的 b_3 作为特例,给出了一个算法去调整 *R-UP* 函数,使得在逻辑上能统一众多的特例.与我们的工作没有关联.

6 结束语

目前对多维数据模型的研究有两个主要的方向:用关系模型表示多维数据模型和把多维数据模型形式化为多维空间.前者不能表示多维数据模型中的一些半结构化特征,后者很难得到广泛的认同.我们提出了用关系对象模型来描述多维数据模型的思想并提出了多维数据模型 $ER(\mathcal{H})$.$ER(\mathcal{H})$的核心思想是把维抽象为一个半结构化对象,可以表示为一个 OEM 图,使用路径表达式作为查询语言.另外,给出了三个完整性规则,扩充了关系代数.可以简洁地表示各种查询.与其它的模型相比,$ER(\mathcal{H})$可以表示更复杂的应用,表达能力更强,可以方便地在主流的关系对象数据库系统上实现.特别地,随着 XML 的普及,许多数据以 XML 文档的形式存在,$ER(\mathcal{H})$模型可以用于 XML 环境.

参 考 文 献

1 Li C., Wang X. S.. A data model for supporting on-line analytical processing. In: Proceedings of the 5th International Conference on Information and Knowledge Management, Rock-

ville, Maryland, 1996, 81～88
2 Agrawal R., Gupta A., Sarawagi S.. Modeling multidimensional databases. In: Proceedings of the 13th International Conference on Data Engineering, Birmingham U.K., 1997, 232～243
3 Gyssens M., Lakshmanan L. V. S.. A foundation for multi-dimensional databases. In: Proceedings of the 23rd International Conference on Very Large Data Bases, Athens, Greece, 1997, 106～115
4 Cabibbo L., Torlone R.. Querying multidimensional databases. In: Cluet Sophie, Hull Richard eds.. Proceedings of the 6th International Workshop on Database Programming Languages. Lecture Notes in Computer Science 1369. Estes Park, Colorado, USA: Springer, 1997, 319～335
5 Lehner W.. Modeling large scale OLAP scenarios. In: Schek Hans-Jörg, Saltor Fèlix, Ramos Isidro, Alonso Gustavo eds.. Proceedings of the 6th International Conference on Extending Database Technology. Lecture Notes in Computer Science 1377. Valencia, Spain: Springer, 1998, 23～27
6 Vassiliadis P.. Modeling multidimensional databases, cubes and cube operations. In: Proceedings of the 10th International Conference on Scientific and Statistical Database Management, Capri, Italy, 1998, 53～62
7 Jagadish H. V., Lakshmanan L. V. S., Srivastava D.. What can hierarchies do for data warehouses? In: Proceedings of the 25th International Conference on Very Large Data Bases, Edinburgh, Scotland, UK, 1999, 530～541
8 Pedersen T. B., Jensen C. S.. Multidimensional data modeling for complex data. In: Proceedings of the 15th International Conference on Data Engineering, Sydney, Austrialia, 1999, 336～345
9 Hurtado C., Mendelzon A. O.. OLAP dimension constraints. In: Proceedings of the Twenty-first ACM SIGMOD Symposium on Principles of Database Systems, Madison, Wisconsin, 2002, 169～179
10 Espil M. M., Vaisman A. A.. Efficient intentional redefinition of aggregation hierarchies in multidimensional databases. In: Proceedings of the 4th ACM International Workshop on Data Warehousing and OLAP, Atlanta, Georgia, 2001, 1～8
11 Lenz H. J., Shoshani A.. Summarizability in OLAP and statistical data bases. In: Proceedings of the 9th International Conference on Scientific and Statistical Database Management, Olympia, Washington, 1997, 132～143
12 Wang H., Zaniolo C.. User defined aggregates in object-relational systems. In: Proceedings of the 16th International Conference on Data Engineering, San Diego, California, 2000, 135～144
13 Abiteboul S., Quass D., McHugh J., Widom J., Wiener J.. The lore query language for semistructured data. International Journal on Digital Libraries, 1997, 1(1): 68～88
14 Huyn N.. Scientific OLAP for the biotech domain. In: Proceedings of the 27th International Conference on Very Large Data Bases, Roma, Italy, 2001, 645～648
15 Gray J., Bosworth A., Layman A., Pirahesh H.. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. In: Proceedings of the 12th International Conference on Data Engineering, New Orleans, Louisiana, 1996, 152～159
16 Zhao Y., Deshpande P., Naughton J. F., Shukla A.. Simultaneous optimization and evaluation of multiple dimensional queries. In: Proceedings of ACM SIGMOD International Conference on Management of Data, Seattle, Washington, 1998, 271～282

LI Sheng-En, born in 1963, Ph. D., professor. His current research interests include data warehousing, OLAP and XML data management.

WANG Shan, born in 1944, professor, Ph. D. supervisor. Her current research interests include database system and knowledge engineering, data warehousing.

Background

This work attributes to the following projects:

"Research on Key Techniques of High Performance Database for Analysis Application", which is supported by the National Natural Science Foundation of China under grant No. 60273017. This project focuses on query optimizing, approximate query processing, data cube compressing and multi-dimensional data model building.

"Non-normal Knowledge Test Platform on Internet", which is supported by the National Natural Science Foundation of China under grant No. 60496325. This project aims at expressing, finding and querying knowledge on Internet.

"Research on Techniques of Domain oriented Data Mining and Data Analysing", which is supported by the National High Technology Research and Development Program of China (863 Program) under grant No. 2002AA4Z3420. The main goal of this project is to build a prototype system which unifying data mining and data analyzing.

This paper proposes a multi-dimensional data model $ER(\mathcal{H})$. It uses Object Exchange Model to unify the member structure and level structure, use path expression to present queries on dimension. Multi-dimension can be as a domain of relation model, and use object oriented and relation data model to model multi-dimensional data. In addition, the research results of XML can be applied to multi-dimensional model.