维普资讯 http://www.cqvip.com

计算机科学1996 Vol. 23 №. 4

基于关系的历史数据库的实现*)

郭 敏 董继润 刘智峰
(山东大学计算机科学系 济南250100)

摘 要 In this paper, we introduce a historical data model. After formalizing the model, we get 2TNF and 3TNF. This can effectively prevent the redundant of data and other abnomal phenomena such as update abnomal. We then discuss the insertions and deletions which maintain the requirement of completeness in the database. A historical relational algebra and its corresponding query language is described in the end.

关键词 Historical databases, RDBMS, Relational algebra, Temporal relational algebra.

时间是现实世界中的一个重要因素。随着数据库技术的发展，越来越多的应用需要保存历史信息，同时光存贮技术也为保存这些信息提供了可能。因此，关于如何在数据库中加入时间的课题在近年来受到了越来越多的重视。

在数据模型中，有可能支持三种类型的时间：有效时间，事务时间，用户定义时间。有效时间指的是现实世界中事件存在的时间。事务时间指把事件存入到数据库中的时间。用户定义时间只是模型中提供来输入、输出或比较的域，顾名思义，它的语义由其应用确定。关于时间的操作，在文[1]中将其归纳为十三种：equals, overlaps, overlapped-by, contains, during, preceeds, follows, before, after, start, end, started-by, ends-by.

传统关系代数只是一种快照代数(Snapshot Algebra)，只支持用户定义时间，而不支持有效时间和事务时间。对关系代数、数据模型而言，若只支持有效时间，则称为历史的；若只支持事务时间，则称为回退的(Roll-back)；若既支持有效时间也支持事务时间，则称为时态的。事务时间比较容易处理，因而本文着重于建立支持有效时间的数据模型和关系代数。

1 历史关系模式的形式化

令 $U_D=\{D_1,\cdots,D_{n_d}\}$是值域的集合($D_i$是一组值)。其中，对每个 D_i, $D_i\neq\varnothing$, $D=\bigcup_{i=1}^{n_d}D_i$ 是所有值的集合。

和每个值域相关的是值比较符集合ΘD_i，其中每个比较符可用来比较域中任意两值的大小。每个比较符集合中至少应包含有=和≠来检查相等或不等。

$TS=\{t_0,t_1,\cdots,t_i,\cdots\}$是一个非空的关于时间的集合，按<排序；是无限可数集。

TIME=[start, end)，其中 start∈TS, end∈TS。

$GS=\{GI_1,GI_2,\cdots GI_{n_g}\}$是关于组标志的集合，组标志的命名可由应用域特征命名。

令 $U_A=\{GS,A_1,A_2,\cdots,A_{n_a}\}$代表属性集合，每个属性由应用域特征命名。其中 GS 为组标志集合，属性 TIME 表示时间，不在 U_A 中。按属性的类型来区分，在 U_A 中的为值属性，TIME 为时间属性。

HR 关系模式 R_{HR}是一个三元组，$R_{HR}=(A,K,DOM)$。其中：

(1)$A\cup(TIME)$是模式 R_{HR}中的属性集合，$A\subset U_A$。

(2)集合 $K\cup(TIME)$是模式 R_{HR}的关键字，其中 $K\subset A$，即 $K\cup(TIME)\rightarrow A$。我们称为关键字限制。

(3)$DOM: A\cup(TIME)\rightarrow U_D\cup\{[TS,TS]\}$是一个函数。每个属性 $A_i\in A$ 有一个值域，用 $DOM(A_i, R_{HR})$表示。U_D 是值域的集合，TIME 的值域为 TS。

HR 关系数据库模式 $DB_{HR}=\{R_{1_{HR}},R_{2_{HR}},\cdots,R_{n_{HR}}\}$是 HR 历史关系模式的一个有限集合。

HR 元组 t_{HR}是 $R_{HR}=(A,K,DOM)$上的一个映射。$t_{HR}: A\cup\{TIME\}\rightarrow D\cup\{[TS,TS]\}$。该映射把每

*)本文得到国家自然科学基金资助

维普资讯 http://www.cqvip.com

个 $A_i \in A$ 的属性 A_i 映射到 $DOM(A_i, R_{HR})$ 上，TIME 映射到 TS 上。

HR 关系 r_{HR} 是模式 $R_{HR}=(A, K, DOM)$ 在 HR 上的历史元组。它们在 R_{HR} 上且满足关键字限制。

HR 数据库 $d_{HR}=\{r_{1_{HR}}, r_{2_{HR}}, \cdots, r_{n_{HR}}\}$ 是 HR 关系元组的集合。每个 r_{HR} 都定义在历史关系模式 R_{HR} 上。

2 历史关系模式的规范化

历史关系模型发展到现在，已经形成了两大分支。一种称为非分组关系(Temporally Ungrouped Relation，以下简称为 TU)，即我们前面定义的 HR。另一种被称为分组关系(Temporally Grouped Relation，以下简称为 TG)，即 N1NF。TG 不同于传统的关系数据库，它允许关系模式中的某些属性的值域中的值是可分解的。

从库的结构来看，TG 比 TU 更为紧凑，容易理解并且 TG 的表达能力强于 TU[4]，但在传统的关系数据库上实现则比较困难，且存在着 N1NF 的种种缺点，如数据冗余，插入异常，删除异常等。J. Clifford 证明了如果在 TU 上加上组标记后，TG 与 TU 是强等价的。因此，我们在 HR 的定义中加入了组标记，HR 与 TG 强等价。

为了减少数据冗余，在文[3]中提到了 TNF 的概念，即在1NF 的基础上给每个时间段加上必须是最大的限制。因而有：任给元组 $R(a_1, a_2, \cdots, a_n, T)$，不会有任何如下的元组存在：$R(a_1, a_2, \cdots, a_n, T')$，其中 $a_1, a_2, \cdots, a_n$ 相等，且时间重叠，即 overlaps(T, T')=TRUE。

采用 TNF 有以下优点：①产生了一个更加紧缩的数据库，因为两个重叠的元组可由一个元组代替。②在非 TNF 中，主键有时会重复。如在只有一个属性 A 的关系 HR 中，A 为主键，我们在库中存了 HR ={(1,[2,10)),(1,[8,20))}，则在8之后主键重复了。③查询语言的实现简单了。

尽管采取了以上措施，我们发现某些关系模式仍然存在问题。我们先来看一个实例。假设职员有以下信息：工资(SALARY)、主管(MANAGER)，性别(SEX)。采用 TG 表示如下：

E#	ENAME	SALARY	MANAGER	SEX
1	Tom Tommy	{(0,4,10K), (4,6,20K) (6,Now,20K)}	{(0,Now,Reter)}	M
2	Scott	{(0,3,20K), (3,7,30K), (7,Now,35K)}	{(0,5,Peter), (5,Now Richard)}	M

转化为 TNF 时采用以下方案：(E#,ENAME,SALARY,MANAGER,SEX)，

E#	ENAME	SALARY	MANAGER	SEX	start	end
1	Tom	10K	Peter	M	0	4
1	Tom	20K	Peter	M	4	6
1	Tommy	20K	Peter	M	6	Now
2	Scott	20K	Peter	M	0	3
2	Scott	30K	Peter	M	3	5
2	Scott	30K	Richard	M	5	7
2	Scott	35K	Richard	M	7	Now

此方案带来了如下问题：

(1)数据冗余　不随时间变化的域—性别，在每条职工记录中都被保存，造成了重复存贮。

(2)更新异常或潜在的不一致性　由于数据存贮冗余，当更新某些数据时，就有可能一部分涉及的元组被修改，而另一部分元组被忽略，从而造成存贮上的不一致性。

(3)删除异常　若删除 Scott 的主管为 Peter 的信息，则将 Scott 在此期间的工资信息也一并删去。

(4)TNF 异常　从整个库上看是 TNF，但从部分信息来看，仍未达到 TNF 的处理思想，造成了存贮浪费。如 $\Pi_{MANAGER, TIME}$HR 不是 TNF，

方案1采用的关系模式是3NF，出现以上异常的原因在于加入了历史信息。

定义2.1　设 HR 是一个历史关系模式，A 为 HR 中的值属性，如果存在 $t_{HR}, t_{HR'}$，使：$\Pi_{GI}t_{HR}=\Pi_{GI}t_{HR'}$，$\Pi_{TIME}t_{HR}\neq\Pi_{TIME}t_{HR'}$ 且 $\Pi_{GI,A}t_{HR}\neq\Pi_{GI,A}t_{HR'}$，则称属性 A 与时间有关。

由此，可将值属性划分为两类。很明显，与时间无关属性的存在是造成数据冗余的主要原因。

定义2.2　设 HR 是一个历史关系模式，HR∈TNF，如果 HR 中每一个属性 A(组标志与时间除外)都与时间有关，则称 HR 是2TNF 的。

在2TNF 中，不存在与时间无关属性。

定义2.3　设 HR 是一个历史关系模式，$X, Y \in U_A$，关系模式 HR 上的时间依赖(Temporal Dependency，简记为 TD)是形为 X→Y 的一个命题，它的含义为：

若存在两个历史元组 t_{HR} 与 $t_{HR'}$，$\Pi_{GI,X}t_{HR}=\Pi_{GI,X}t_{HR'}$

且 follows$(\Pi_{TIME}t_{HR}, \Pi_{TIME}t_{HR'})$=TRUE

则 $\Pi_{GI,Y}t_{HR}=\Pi_{GI,Y}t_{HR'}$

属性之间存在非时间依赖是造成 TNF 异常的主要原因。

定义2.4　设 HR 是一个历史关系模式，如果 TD$\subset U_A$ 且 TD 中任两个属性 X，Y，都有 X→Y，则称 TD 是 HR 的一个完全时间依赖集。

维普资讯 http://www.cqvip.com

定义2.5 设HR是一个历史关系模式，如果TD是HR的一个完全时间依赖集，若不存在$X\in A_U, Y\in TD$，使$X\rightarrow Y$且$Y\rightarrow X$，则称TD是HR的一个最大完全时间依赖集。

定理2.1 设HR是一个历史关系模式，$X,Y,Z\in U_A$，若有$X\rightarrow Y, Y\rightarrow Z$，则$X\rightarrow Z$。

定义2.6 设HR是一个历史关系模式，如果$HR\in 2TNF$且对HR中任意两个属性$X,Y\in U_A$，都有$X\rightarrow Y$，则称HR是3TNF的。

在3TNF中，由于各属性的变化同步，因而不存在TNF异常的问题。

下面讨论将N1NF分解为3TNF。

N1NF中有四种类型的属性[2]，它们是：①简单属性(simple valued)；②集合属性(set valued)；③三元组属性(triplet valued)；④三元组集合属性(set triplet valued)。如在前面的例子中，E#和ENAME是简单属性，SALARY是三元组集合属性。因为N1NF是GR(分组关系)，所以其中必有属性集K_G，将N1NF分为若干组，称K_G为分组关键字。

算法：将N1NF分解为3TNF

输入：一个历史关系模式$R_{HR}=A_1,\cdots,A_n$，R_{HR}的所有最大完全时间依赖集TD_i的集合TD，R_{HR}是N1NF。

输出：R_{HR}的一个分解$P=(R_1,\cdots,R_K)$，且每个R_i是3TNF。

(1)$R_1=\{GI,K_G,TIME\}$，$U_A'=U_A-GI-K_G$；

(2)对U_A中的所有简单属性$A_{s_1},\cdots,A_{s_m}$建$R_2=\{GI,A_{s_1},\cdots,A_{s_m}\}$；

$U_A'=U_A-\{A_{s_1},\cdots,A_{s_m}\}$

(3)对每个完全时间依赖集$TD_i\in TD$，建R_{2+i}

for i=1 to n do

$R_{2+i}=\{GI,TD_i,TIME\}$；

$U_A'=U_A-TD_i$；

enddo；

(4)对每个属于U_A的属性$A_{T1},\cdots,A_{Tp}$，建R_{2+n+j}

for j=1 to p do

$R_{2+n+1}=\{GI,A_{Tj},TIME\}$；

$U_A'=U_A-\{A_{Tj}\}$；

enddo；

3 在HDB中实现插入，删除

3.1 插入HDB

为保证存在RDB中的时间段是最大的，必须不是简单地往库中加入新元组，重叠的元组应予合并。R中元组$t_{New}=(T_{New},a_1,\cdots,a_n)$的加入由以下Procedure实现。

procedure insertHDB(HR,t_{New})

(1)$S_1=(T')$ for-which$(T',a_1',\cdots,a_n')\in HR$ and $a_1'=a_1,\cdots,a_n'=a_n$)and overlaps(T_{New},T')=TRUE；

(2)for-all $t=(T',a_1',\cdots,a_n')$且$T'\in S$ delete t from HR；

(3)$S_1=S\cup\{T_{New}\}$；

(4)start$(T_{Result})'$=mininum{start(T'), for-all $T'\in S$}，end$(T_{Result})'$=maximum{end(T'), for-all $T'\in S$}；

(5)insert$(T_{Result},a_1,\cdots,a_n)$into HR.

第一步：查找在关系中已存的元组是否在加入t_{new}后损害了TNF的定义(TNF：时间间隔必须最大)，这些元组具有与t_{new}相同的属性且时间间隔与T_{new}相重叠，查到后，放入S中；

第二步：将所有第一步中的元组删除；

第三步：把T_{new}加入到集合S中；

第四步：计算包括已有元组和T_{new}的最大间隔；

第五步：插入新元组，该元组包括t_{new}的值属性和T_{Result}。

3.2 从HDB中删除

从HDB中删除一段时间的元组，同样需要修剪所有库中与之重叠的时间段，算法略。

4 历史关系代数

4.1 选择

因为σ只是查找关系中的元组，所以若关系R是TNF，则输出结果也必是TNF，因此可以直接使用RA的σ来实现HRA的σ。

4.2 投影

若关系R是3TNF，则投影的结果必为TNF。若R只是TNF，实现投影时，就需要在输出中维持TNF，因为不同的元组经过投影后可能变成相同元组，因而也就破坏了TNF。如从EMP中投影NAME属性会产生以下结果：

$\Pi_{NAME,TIME}(\sigma_{NAME='Tom'}EMP)=\{(Tom,[0,4)),(Tom,[4,6))\}$

该结果显然不符合TNF的定义，因此使用RA的投影以后还要利用coalesce算法[4]合并那些内容相同、时间重叠的元组。因而，投射有以下形式：

$\Pi_{t.x,y,\cdots}HR=coalesce(\Pi_{x,y,\cdots,TIME}HR)$；

4.3 差

时间段的结构使得HR－HS有可能产生额外的元组，如：{(Scott,30K,[3,5))}－{(Scott,30K,[4,4)}生成了两个元组：{(Scott,30K,[3,3)},{(Scott,

维普资讯 http://www.cqvip.com

30[5,5)}。

除保留原有的非历史关系的差运算外，再增加历史关系的差运算－t，其实现由以下过程完成：

```
FUNCTION difference(HR,HS)
  HR':=HR ORDER-BY A1,…,An,TIME;
  HS':=HS ORDER-BY A1,…,An,TIME;
  size:=COUNT(HR);Rd:=0;
  REPEAT
    (a1,…,an,T):=HR';
    DO WHILE EXISTS(a1',…,an',T')=HS and
      a1=a1',…,an=an' and overlaps(T,T')=TRUE
      IF after(T,T')=TRUE THEN
      T':=[end(T'),end(T));/*T 发生在 T' 的 start
          之后且有重叠*/
      ELSE IF contains(T,T')=TRUE THEN
        T'':=[start(T),start(T'));/*T 包含 T',则 T
            应分为两个时间段*/
      INSERT(a1,…,an,T'')INTO Rd;
      T:=[end(T'),end(T));
      ELSE IF before(T,T')=TRUE THEN
      T:=[start(T),start(T'));/*T 发生在 T' 的 start
          之前且有重叠*/
      ELSE IF contains(T',T)=TRUE THEN
      T:=NULL;/*T' 包含 T,则不插入任何记录*/
      END IF;
    ENDDO;
    IF T!=NULL THEN INSERT(a1,…,an,T)INTO Rd;
      END IF;
    i:=i+1;
  WHILE i≤size;
  RETURN Rd;
END difference;
```

注意：以上程序中若 T＝[start，end)且 start＝end，则 T 默认为 NULL。因而，历史关系下的差运算有以下形式：

HR－tHS＝difference(HR，HS)。

4.4 并

与投影一样，并也会产生一个破坏 TNF 的结果，因此我们用以下函数来合并结果的重叠部分。HR∪tHS＝coalesce(HR，HS)。

4.5 乘积

历史关系的乘积最为复杂，乘积是破坏3TNF的主要原因之一。以下使用了三种乘积来实现适用于时间结构的操作：

①Tempotal-Natural-Product：R⋈tS，若 R 与 S 都是 HR，则 TIME 为：

[maximum(start(r_R)，start(r_S))，minimum(end(r_R)，end(r_S)))

②Since-Product：R×sS，R 中元组与 S 中元组的乘积，若 R 与 S 都是 HR，则 TIME 为：

[maximum(start(r_R)，start(r_S))，end(r_R))

R×sS 的结果中，若对 R 中的属性投影，则得到的结果与 R 一样。但若对 S 中的属性投影，则 S 是得到的结果的子集。该运算对与下例类似的查询非常有用。

例：Scott 在工资为30K 以后的主管是谁？

$\Pi_{t\text{MANAGER}}$ ($\sigma_{\text{SALARY}=30K}$ (TMANAGER ×sTSALARY)

③Until-Product：R×uS，R 中元组与 S 中元组的乘积，若 R 与 S 都是 HR，则 TIME 为：

[minimum(start(r_R)，start(r_S))，end(r_S))。

5 历史查询语言 HQL

5.1 句法

项是一个常数或变量，若常数和变量与时间有关，则称之为时态项。设 $t_1,\cdots,t_N$ 是项，T 是时态项、p 是一个 n 维谓词，则 $p(t_1,\cdots,t_N)@T$ 是时态原子。其含义是指 $p(t_1,\cdots,t_N)$在时间段 T 内是有效的，T 被称为该时态原子的有效时间参数。对∃ T(sick(Emp)@T)可简写为 sick(Emp)。

除了固有的谓词(＜、＞、≤、≥、＝、≠)以外，我们还引进了文[1]中提到的十三种时态嵌入谓词(before，after，…)。

每个时态原子是一个时态公式，另外，若 F_1，F_2 是时态公式，x 是变量，T 是时态项，则 $F_1\vee F_2$，$F_1\wedge F_2$，$\neg F_1$，$F_1\rightarrow F_2$，∃ xF_1，∀ xF_1 和 $\{F_1\}@T$ 都是时态公式。其中 $\{F_1\}@T$ 中的大括号代表 F_1 应按时间来衡量。同样∃ T{F}@T 和{F}是等价的。

时态演绎数据库包括了有限的时态事实和时态规则。时态事实是一个时态项、时态规则是形如 H←B 的公式，H 是项而 B 是时态公式。H 被称为规则头，B 为规则体。一个时态查询形如?-Q，其中 Q 是时态公式。

5.2 语义

时态公式{φ}@[start，end)为真，当且仅当：

1. φ 在任何时间点 t 为真，且 start≤t＜end。
2. φ 在时间点 end 处为假。
3. φ 在 start 以前的时间点为假。

以上定义中的2和3保证了[start，end)是最大的，即为 TNF。

以下的规则规范了在时间点 t 为真这一概念：

I_t (p(x)@[start，end))为真，若 p(x)@[start，end)为真且 start≤t＜end。

I_t (p(x))为真，若 p(x)@[start，end)为真且存在某些 t，start≤t＜end。

$I_t(\varphi\wedge\psi)$为真，若 $I_t(\varphi)$且 $I_t(\psi)$为真。

$I_t(\varphi\vee\psi)$为真，若 $I_t(\varphi)$或 $I_t(\psi)$为真。

$I_t(\neg\varphi)$为真，若 $I_t(\varphi)$为假。

I_t(∃ $x\varphi$)为真，若对于某些 c，$I_t(\varphi[x/c])$为真。

I_t(∀ $x\varphi$)为真，若对于所有 c，$I_t(\varphi[x/c])$为真。

下面给出了 HQL 的语义及其对应的历史关系表达式，这种对应只是非正式的，如本应使用矢量参数的地方我们用一个或两个参数来替代，这可以使我们更好地注意到有效时间参数的表达和变换。

维普资讯 http://www.cqvip.com

HQL 表达式	相应的历史关系代数表达式
{p(x)∧q(x,y)}@T	$\Pi_{t_{s,e}}(P' \bowtie_t Q')$
{p(x)∨q(x)}@T	$\Pi_t(P' \cup_t Q')$
{p(x)∧¬q(x)}@T	$\Pi_{t_s,t_e}(P' -_t Q')$
{∃ y p(x,y)}@T	$\Pi_{t_s,t_e}(P')$
{p(x)∧f(x)}@T	$\Pi_{t_s,t_e}(\sigma_{F,t_s,t_e}P')$

例:有以下两个库:

employee(职工):

E#	Salary	validTime	
		start	end
01	4000	1993/05/01	1993/07/01
01	4200	1993/07/01	1994/05/01
02	5250	1993/08/01	1994/02/01
03	4300	1993/08/01	1993/02/01
03	4600	1993/02/01	1993/09/01
04	5000	1993/11/01	1994/05/01

manager(管理人员):

MgrE#	EmpE#	validTime	
		start	end
03	01	1993/05/01	1993/08/01
02	01	1993/08/01	1993/11/01
04	01	1993/11/01	1994/05/01

查询是:当03管理01时的工资是多少以及接替03管理01的职员的工资是多少?

```
?-{employee('03',sal-03)&
   manager('03','01')}@ValidTime1&
   {manager(who,'03')&
    employee(who,sal-who)}@ValidTime2 &
   ValidTime2 follows ValidTime1.
```

此查询包括两个子查询.第一个查出了03管理01时的工资,每二个查出了管理01的职员的编号和工资。最后将两个子查询合并并使它们的有效时间满足给定条件,此查询产生的结果是:

Sal_03	ValidTime1		Who	Sal_Who	ValidTime2	
	start	end			start	end
4600	1993/05/01	1993/08/01	02	5250	1993/08/01	1993/11/01

6 总结与展望

本文对历史数据模型进行了形式化和规范化、并提出了2TNF 和3TNF 的概念。从而解决了 TNF 中存在的数据冗余等问题、同时提供了一个将 N1NF 分解为3TNF 的算法.在此基础上,介绍了在 3TNF 中保持数据完整性的前提下数据的插入和删除操作。最后提供了历史关系代数和相应的历史查询语言。

时间数据库发展到现在,大部分工作都是围绕在关系数据库上实现展开的,由于时间的特有语义和操作,使之很符合对象的特点,它的简单定义如下:

```
CALSS TIME
{ATTRIBUTE
   start
   end
 METHOD
   equals
   overlaps
   ...
}
```

从这个意义上讲,历史数据库可以看成是复杂对象的集合,每个复杂对象包括与时间有关的属性组成的对象和 TIME,因而在面向对象数据库中实现时间数据库将是以后工作的重点。

参考文献

[1]J. F. Allen, Maintaining Knowledge about Temporal Interval, CACM, 16(11), 1983

[2]A. U. Tansel 等, Time_by_Example Query Language for Historical Databases, IEEE Trans. on Software Engineering, 15(4), 1989

[3]P. McBrien, Principles of Implementing Historical Databases in RDBMS, Advances in Databases, Lecture Notes in Computer Science, 696, 1993

[4]J. Clifford 等, On Completeness of Historical Relational Query Languages, ACM Trans. on Databases Systens, 19(1), 1994

[5]周龙骧,《数据库管理系统实现技术》,中国地质大学出版社,1989

[6]M. Bohlen, R. Marti, On the Completeness of Temporal Database Query Languages, Temporal logic, Lecture Notes in Artificial Intelligence, 827, 1994