数据仓库中时态视图的维护*

李 琪[1], 白英彩[2]

[1](上海电信公司 帐务中心,上海 200071);
[2](上海交通大学 计算机科学与工程系,上海 200030)
E-mail: happlq@263.net
http://www.sjtu.edu.cn

摘要: 数据仓库的一个重要用途是利用时态视图向用户提供历史信息.因为在传统关系数据模型中增加了对时间的支持,而且时态视图的更新不仅来自于基表更新,还包括时间前进,所以,目前对非时态视图维护的研究成果不适用于时态视图,并且已有的一些时态视图维护算法也不适用于数据仓库.以历史关系模式为对象,根据增量式维护方法的原理,采用纯删除、纯插入的计算方法,用代数语言给出了 5 种基本历史关系代数运算的更新传播算法,由这 5 种历史关系代数组合定义的时态视图都可用迭代方法得到其增量维护计算式.所采用的纯删除、纯插入思想也可移用于其他历史关系模式下的时态视图维护.

关 键 词: 数据仓库;时态操作;视图维护;历史关系模式;增量式计算
中图法分类号: TP311 文献标识码: A

数据仓库为人们进行信息查询和决策分析提供了良好的综合数据源,保存历史数据、支持历史查询是数据仓库的重要任务.为了支持历史查询,定义涉及时间属性的视图是必不可少的,并将其称为时态视图(temporal view).为了加快查询速度,通常将查询频率高的视图内容保存在数据仓库中,称之为实体化视图(materialized view),对实体化视图内容如何更新就是视图维护所要解决的问题.在一般的关系数据库中,将时间属性与其他属性同等对待,不支持时间比较、时间语义等时间相关特性,这给书写历史查询和时态视图的维护带来不便,必须通过特定程序才能完成.因此,在很多应用中采用支持时间属性的时态数据库对于数据仓库来讲是必要的,而且通常从数据仓库的使用需求来说,支持有效时间采用历史关系数据库就足够了.

视图维护可以采用重新计算和增量式计算这两种方法,重新计算是用更新后的基表从头计算视图,不再使用视图中原来的结果.而增量式更新只是计算出视图的更新部分,用它去更新旧视图得到新视图,没有改变的那部分视图不必再重新计算得到.实验证明,当基表更新只占基表较小比例时,增量式维护比重新计算的性能要好.对于增量式视图维护的研究已经取得了很多成果[1],但对于时态视图的维护研究还不多.文献[2]研究了在 chronicle 数据模型下时态视图的维护问题,在此数据模型中每个元组都带有一个时间戳标记,但只是单个时间值而不是时间段;文献[3,4]提出了增量式时态视图维护技术,但它并不是定义于整个历史的时态视图,而是用非时态操作符定义于若干个基表快照上的视图,为了访问整个事件历史必须显式使用 backlog 说明符,文中还使用了 differential 操作用于更新传播;文献[5]考虑了在选择条件中用到 NOW 的关系查询维护,但其每个元组只有一个时间戳标记.以上这些文献中的算法针对的都不是规范的历史数据模型,限制了时态查询的表达能力.文献[6]给出了一个在历史数据模型下的时态视图维护算法,但是它所针对的历史数据模型不易于在传统的关系数据库中实现,同时这种维护算法不能迭代产生一个由多个代数操作组成的查询表达式的维护算法,未将基表更新和时间前进对视图的影响综合考虑,而是分开完成.为此,本文在基于关系的历史数据库模型下,提出时态视

* 收稿日期: 2000-09-28; 修改日期: 2001-03-12

作者简介: 李琪(1973－),女,重庆人,博士生,主要研究领域为数据仓库,视图维护;白英彩(1936－),男,辽宁沈阳人,教授,博士生导师,主要研究领域为计算机网络,分布式计算.

图的增量式维护算法,较好地克服了以上缺点.

1 历史关系模式

1.1 历史关系模式的形式化

$TS=\{t_0,t_1,\ldots,t_i,\ldots\}$是一个非空的关于时间常量的集合,按照<排序,是无限可数集,且 $P_{TS}=\{c|c\subseteq TS\cup\{\text{NOW}\}\wedge c\neq\varnothing\}$,其中字符 NOW 表示当前时间,它的值随时间前进不断变换.

时间属性定义为 T=[start,end),其中 start$\in P_{TS}$,end$\in P_{TS}$,且 start≤end.

令 $U_A=\{A_1,A_2,\ldots,A_{n_A}\}$ 代表属性集合,每个属性由应用域特征命名,属性 T 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,T 为时间属性.

历史关系模式 R_{HR} 是一个三元组,R_{HR}=〈A,K,DOM〉.其中

(1) $A\cup\{T\}$是此模式的属性集合,$A\subseteq U_A$.

(2) 集合 $K\cup\{T\}$是模式的关键字,即 $K\cup\{T\}\rightarrow A$,我们称为关键字限制,其中 $K\subseteq A$.

(3) $DOM: A\cup\{T\}\rightarrow U_D\cup\{[P_{TS},P_{TS})\}$ 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,T 的值域为 P_{TS}.

历史关系 r_{HR} 是模式 R_{HR}=〈A,K,DOM〉上满足关键字限制的有限元组集合.

一个历史关系元组 t_{HR} 是 R_{HR}=〈A,K,DOM〉上的一个映射. $t_{HR}: A\cup\{T\}\rightarrow U_D\cup\{[P_{TS},P_{TS})\}$.该映射把每个 $A_i\in A$ 的属性 A_i 映射到 $DOM(A_i,R_{HR})$ 上,T 映射到 P_{TS} 上.

为了减少数据冗余,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 $R(a_1,a_2,\ldots,a_n,T)$,不会有任何如下的元组存在 $R(a_1,a_2,\ldots,a_n,T')$,其中 $a_1,a_2,\ldots,a_n$ 相等且有效时间 T 与 T'重叠.

1.2 历史关系代数

在这种历史关系模式中,要对传统的 5 种基本关系代数:差(difference)、并(union)、投影(projection)、选择(selection)和连接(join)进行时态扩充.在下列描述中,R,S 是历史关系,r,s 分别是这些关系中的元组, A_R, A_s 分别表示关系中的值属性集合,不包括时间属性.令 $r.A$ 表示元组 r 中其值属性在 A 中的属性值.定义函数 $h(R,r',t',A)=\{r.T\mid\exists r\in R: r.A=r'.A\wedge r.T\cap t'\neq\varnothing\}$.

• selection $\sigma_p(R)=\{r\mid r\in R\wedge p(r)\}$,其中 p 是选择条件.

• projection $\pi_A(R)=coalesce(\pi^{nt}_A(R))$,其中 $\pi^{nt}_A(R)$ 表示传统非时态关系投影操作,coalesce 算法合并那些非时间属性值相同、时间重叠的元组.

• difference $R-S=\{\langle r.A_R,c\rangle r\in R\wedge c=r.T-h(S,r,r.T,A_R)\wedge c\neq\varnothing\}$.注意,只在 A_R=A_S 时进行差操作.

• union $R\cup S=coalesce(R\cup^{nt}S)$,与投影运算类似,其中 $\cup^{nt}$ 表示非时态关系并操作,coalesce 算法合并那些非时间属性值相同、时间重叠的元组.注意,只在 A_R=A_S 时进行并操作.

• join $R\rhd\lhd_p S=\{\langle r.A_R,s.A_S,c\rangle\mid r\in R\wedge s\in S\wedge p(r,s)\wedge c=r.T\cap s.T\wedge c\neq\varnothing\}$.

1.3 时间操作

Table 1 The definition of temporal operation

表 **1** 时间操作定义

Temporal operation①	Mathematics expression②
equals (α,β)	$B(\alpha)=B(\beta)\wedge E(\alpha)=E(\beta)$
overlaps (α,β)	$B(\alpha)<B(\beta)\wedge E(\alpha)\geq B(\beta)\wedge E(\alpha)<E(\beta)$
OVERLAPS(α,β)	$(B(\alpha)\geq B(\beta)\wedge B(\alpha)\leq E(\beta))\vee(B(\alpha)\leq B(\beta)\wedge E(\alpha)\geq B(\beta))$
contains (α,β)	$B(\alpha)<B(\beta)\wedge E(\alpha)>E(\beta)$
meets (α,β)	$E(\alpha)+1=B(\beta)$

①时间操作,②数学表达式.

对于时间操作,文献[7]将其归纳为 13 种 equals,overlaps,overlapped-by,contains,during,proceeds,follows,before,after,start,end,started-by,ends-by.表 1 只给出本文要用到的时间操作的定义,另外增加一个广义时间重叠

操作 OVERLAPS.设α,β是两个时间间隔,函数 B,E 分别获取时间间隔的起始时间和结束时间.

2 时态视图的维护

2.1 时态视图维护实例

设表 2 为历史关系 R 的实例.我们定义视图 $V=\pi_{\{ENAME,SALARY,start,end\}}(\sigma_{(start\geq NOW-5year)}R)$.在 1998 年 12 月 25 日,关系的内容见表 2,视图 V 的内容见表 3,关键字 NOW 表示当前时间.

在 1999 年 1 月 3 日,基表 R 中插入元组(TOM,15K,06/01/97,01/01/99),(TOM,20K,01/01/99,NOW),那么此时再重新计算视图 V 得到的结果见表 4.

从表 4 的结果来看,是从表 3 中删除了第 1 个元组,修改了第 3 个元组又增加了一个新元组.其中第 3 个元组的修改是由于基表 R 中新元组(TOM,15K,06/01/97,01/01/99)的插入,第 1 个元组的删除是由于此元组已经不满足当前的时间选择条件,而插入的新元组是由于基表中插入的新元组(TOM,20K,01/01/99,NOW)符合视图当前时间选择条件.

Table 2

表 2

ENAME	SALARY	Start	End
TOM	10K	01/01/91	05/01/92
TOM	20K	01/01/94	06/01/96
TOM	10K	06/01/96	12/31/96
TOM	15K	06/01/97	NOW

Table 3

表 3

ENAME	SALARY	Start	End
TOM	20K	01/01/94	06/01/96
TOM	10K	06/01/96	12/31/96
TOM	15K	06/01/97	NOW

Table 4

表 4

ENAME	SALARY	Start	End
TOM	10K	06/01/96	12/31/96
TOM	15K	06/01/97	01/01/99
TOM	20K	01/01/99	NOW

从这个更新实例我们看到,时态视图有以下两点不同于传统的非时态关系视图维护:

(1) 单调性的失去.有的历史关系代数操作并不像其对应的关系操作那样具有单调性,比如上例中的选择操作,即我们不能保证 $op(...,R,...)\subseteq op(...,R\cup R',...)$,而在所有关系操作中除了差运算以外都具有单调性,它为实现增量式视图维护提供了很好的基础,因为我们可以断定,基表的插入只会对视图造成插入而不是删除或修改.而在时态视图的维护中,我们不能运用单调性,因为 $R\cup R'$ 会造成对时间属性的修改而不仅仅是新元组的插入,$R-R'$ 同理.

(2) 时间前进对视图的影响.从实例中我们看到,在 1998 年 12 月 25 日,R 中的元组(TOM,20K,01/01/94,06/01/96)符合视图条件,而到了 1999 年 1 月 3 日,这个元组就不满足视图条件而必须从视图中删除,这是因为视图定义的选择条件中有关键字 NOW,它的值随着时间的前进而不断改变.相反,还有另一种情况,基表中有些元组原来不满足视图条件,但随着时间前进,它在某个时候满足视图条件而必须插入视图,所以说随着时间前进也会引起视图中元组的插入或删除.这正是人们所渴望的时态运算的一个重要性质,即用户可以看到即时更新的视图又无须定制的编程.

2.2 时态视图的维护

2.2.1 基本思想

假设 $Q(R_1,\ldots,R_n)$ 是一个定义时态视图的历史查询,其中 $R_1,\ldots,R_n$ 是历史关系,由前面例子可知,对历史关系表的插入(删除)会引起表中元组的修改,而使历史关系代数操作失去了单调性,为此我们将这种修改转换为一对删除-插入操作,使得这些删除(插入)元组不会对历史关系产生修改,称其为纯删除(纯插入),但基表的纯删除(纯插入)对时态视图的影响也有可能是修改,为此在更新传播中也用纯删除(纯插入)计算更新元组.采取这种纯删除(纯插入)的另一个优点是可以缩短视图锁定时间.因为在视图作更新修改时是不能为用户提供查询访问的,所以应该在视图维护前尽量多地完成一些必须的工作,以减轻视图维护期间的计算负担,缩短维护时间.

假设关系 R_i 在时间由 t_1 前进到 t_2 的这段时间里,待插入(删除)的元组是 ∇R_i 和 ΔR_i,将 ∇R_i 和 ΔR_i 转换为纯删除 $\nabla' R_i$ (纯插入 $\Delta' R_i$),则

$$R_i \quad \text{at} \quad t_2 = R_i \quad \text{at} \quad t_1 \Theta \nabla' R_i \oplus \Delta' R_i, i \in (1,\ldots,n), \tag{1}$$

式(1)中的 Θ 和 $\oplus$ 分别表示纯删除和纯插入操作,$\nabla' R_i$ 和 $\Delta' R_i$ 满足

$$\left.\begin{aligned} \nabla' R_i \cap (R_i \quad \text{at} \quad t_1) \equiv \nabla' R_i, \\ \Delta' R_i \cap (R_i \quad \text{at} \quad t_1) \equiv \varnothing. \end{aligned}\right\} \tag{2}$$

注意,式(2)中的∩表示的是逻辑运算而非历史关系运算.

从上述定义可得出,$\nabla' R_i \cap \Delta' R_i \equiv \varnothing$,也就是说,它们相互之间没有任何交叉参考,不会给视图维护造成多余的计算,这也是一个好的增量式维护算法所需满足的性质之一.若历史查询 Q 只包含一个历史关系代数操作 op,其更新传播公式为

$$\left.\begin{aligned} op(R_1,\ldots,R_i - \nabla R_i \ldots,R_n)\text{at} \quad t_2 = op(R_1,\ldots,R_i,\ldots,R_n)\text{at} \quad t_1 \Theta \nabla^- \varepsilon_{R_i} \oplus \Delta^- \varepsilon_{R_i}, \\ op(R_1,\ldots,R_i \cup \Delta R_i \ldots,R_n)\text{at} \quad t_2 = op(R_1,\ldots,R_i,\ldots,R_n)\text{at} \quad t_1 \Theta \nabla^+ \varepsilon_{R_i} \oplus \Delta^+ \varepsilon_{R_i}. \end{aligned}\right\} \tag{3}$$

其中 $\nabla^- \varepsilon_{R_i}$ 和 $\Delta^- \varepsilon_{R_i}$ ($\nabla^+ \varepsilon_{R_i}$ 和 $\Delta^+ \varepsilon_{R_i}$)是可能包含 $R_1,\ldots,R_n$,∇R_i (ΔR_i)的表达式,分别表示旧视图 $op(R_1,\ldots,R_n)$ 中的待删除、待插入元组,而且从对上节例子的分析可知,时态视图的更新来源于两方面:基表更新和时间的前进,所以 $\nabla^- \varepsilon_{R_i}$ 和 $\Delta^- \varepsilon_{R_i}$ ($\nabla^+ \varepsilon_{R_i}$ 和 $\Delta^+ \varepsilon_{R_i}$)中包含这两方面的作用成分.如果对于所有的历史关系操作,我们都已经得到其相应的 $\nabla^- \varepsilon_{R_i}$ 和 $\Delta^- \varepsilon_{R_i}$ ($\nabla^+ \varepsilon_{R_i}$ 和 $\Delta^+ \varepsilon_{R_i}$),那么对于任何一个组合历史查询 Q(包含一个以上历史关系操作),我们都可以运用迭代的方法重复使用式(4),得到形如下式的视图更新计算公式:

$$Q \text{ at } t_2 = Q \text{ at } t_1 \Theta \nabla Q \oplus \Delta Q. \tag{4}$$

下面分两步计算这 5 种基本历史关系操作的更新传播,第 1 步先考虑基表更新,第 2 步则在基表更新的基础上加入时间前进的影响.

2.2.2 基表更新

本文采取直接寻找法,首先确定视图中会被基表更新修改的元组并将之删除,接着再计算要插入的元组,这样得到的删除(插入)元组就不会对视图造成修改操作,而且可以使用非时态插入和删除的计算方法.

先定义一个时态半连接操作符 $\propto_p : R \propto_p T = \{r \mid \exists r,t : r \in R \wedge t \in T \wedge p(r,t)\}$.

(1) 基表 R: 设 ∇R (ΔR)分别为基表 R 中待删除(插入)的元组,可根据下式将之转换为纯删除(纯插入):

$$\nabla^- \varepsilon = R \propto_{OVERLAPS(\$1.T,\$2.T) \wedge (\$1.A_1 = \$2.A_2)} \nabla R, \Delta^- \varepsilon = \nabla^- \varepsilon - \nabla R, \tag{5}$$

$$\nabla^+ \varepsilon = R \propto_{OVERLAPS(\$1.T,\$2.T) \wedge (\$1.A_1 = \$2.A_2)} \Delta R, \Delta^+ \varepsilon = \nabla^+ \varepsilon \cup \Delta R, \tag{6}$$

式中,$\$1.T$ 表示参与关系运算的第 1 个关系的元组在时间属性上的值,$\$1.A_1$ 表示参与关系运算的第 1 个关系的元组在非时间属性上的值,$\$2.T$ 和 $\$2.A_2$ 对应的是第 2 个关系元组,以下相同.为描述方便,就用 ∇R, ΔR 分别表示基表 R 的纯删除和纯插入.

(2) 投影操作 $\pi_A(R)$:投影操作的处理比较复杂,因为当投影属性中包含时间属性时,投影后要合并那些非时间属性值相同、时间重叠的元组,因此有

$$\left.\begin{aligned} \nabla^- \varepsilon_R = \pi_A(R) \propto_{OVERLAPS(\$1.T,\$2.T) \wedge (\$1.A = \$2.A)} \nabla R, \\ \Delta^- \varepsilon_R = \pi_A(R \propto_{(\$1.A = \$2.A) \wedge \neg equals(\$1.T,\$2.T)} \nabla R) - \pi_A(R). \end{aligned}\right\} \tag{7}$$

$$\left.\begin{aligned} \nabla^+ \varepsilon_R = \pi_A(R) \propto_{OVERLAPS(\$1.T,\$2.T) \wedge (\$1.A = \$2.A)} \Delta R, \\ \Delta^+ \varepsilon_R = \pi_A(R \propto_{(\$1.A = \$2.A) \wedge OVERLAPS(\$1.T,\$2.T)} \Delta R \cup \Delta R). \end{aligned}\right\} \tag{8}$$

式(7)和式(8)中 A 是投影属性不是基表 R 的值属性.

(3) 并操作 $R \cup T$:与投影操作类似,需要对非时间属性值相同、时间重叠的元组进行合并.

$$\left.\begin{aligned} \nabla^- \varepsilon_R = (R \cup T) \propto_{OVERLAPS(\$1.T,\$2.T) \wedge (\$1.A_1 = \$2.A_2)} \nabla R \Theta \nabla R_{equals(\$1.T,\$2.T) \wedge (\$1.A_1 = \$2.A_2)} T, \\ \Delta^- \varepsilon_R = \nabla^- \varepsilon_R - \nabla R \propto_{(\$1.A_1 = \$2.A_2) \wedge \neg equals(\$1.T,\$2.T)} T. \end{aligned}\right\} \tag{9}$$

$$\left.\begin{aligned}\nabla^+\varepsilon_R &= (R\cup T)\propto_{OVERLAPS(\$1.T,\$2.T)\wedge(\$1.A_1=\$2.A_2)}\ \Delta R,\\ \Delta^+\varepsilon_R &= \Delta R\cup\nabla^+\varepsilon_R.\end{aligned}\right\} \tag{10}$$

$\nabla^+\varepsilon_R$ 是找出旧视图中所有会与待插入元组发生合并的元组; $\Delta^+\varepsilon_R$ 的形成一部分是因为 ΔR 与视图中的旧元组发生合并产生的,另一部分则是 ΔR 中不能与旧元组合并的元组.

由于 R 和 T 存在着对并操作的同等性,只要将式(9)和式(10)中的 R,T 位置对调就得到 T 的相应更新传播计算式.

(4) 差操作 $R-T$:时间段的结构使得 $R-T$ 有可能产生额外的元组,如{(Scott,30K,[3,8])}−{(Scott,30K,[4,5])}生成了两个元组:{(Scott,30K,[3,4])}和{(Scott,30K,[5,8])}.参与差操作的两个关系 R,T 其更新传播公式是不一样的,首先考虑 R 的更新传播计算式:

$$\nabla^-\varepsilon_R = (R-T)\propto_{OVERLAPS(\$2.T,\$1.T)\wedge(\$1.A_1=\$2.A_2)}\nabla R, \Delta^-\varepsilon_R=\varnothing, \tag{11}$$

$$\Delta^+\varepsilon_R = (\Delta R-T), \nabla^+\varepsilon_R=\varnothing. \tag{12}$$

因为 ΔR 是纯插入的,它的时间属性值与 R 中的任何一个元组没有重叠部分,而 $R-T$ 中每个元组的时间属性值是对应 R 中元组的时间属性值的子集,所以旧视图中不存在因 ΔR 的插入而被修改的元组.同样也没有因为 ∇R 的删除而被修改的元组,虽然 $\nabla R-T$ 可以看做是旧视图中待删除的元组,但更简单的方法是从旧视图中找出那些与 ∇R 中元组非时间属性值相同而时间属性值有重叠的元组.

T 的更新传播计算式为

$$\left.\begin{aligned}\nabla^-\varepsilon_T &= ((R-T)\propto_{meets(\$1.T,\$2.T)\wedge(\$1.A_1=\$2.A_2)}\nabla T)\propto_{\neg equals(\$1.T,\$2.T)\wedge(\$1.A_1=\$2.A_2)} R,\\ \Delta^-\varepsilon_T &= (\nabla^-\varepsilon_T\cup(\nabla T\propto_{meets(\$1.T,\$2.T)\wedge(\$1.A_1=\$2.A_2)}\nabla^-\varepsilon_T))\oplus(R\propto_{(\$1.A_1=\$2.A_2)\wedge equals(\$1.T,\$2.T)}\nabla T).\end{aligned}\right\} \tag{13}$$

$\nabla^-\varepsilon_T$ 是 R 减去 ∇T 后的结果元组,根据减操作的定义可知,这些结果元组的时间属性值与被减元组的时间属性值存在着首尾时间相同的关系.待插入的元组 $\Delta^-\varepsilon_T$ 分为两部分,第 1 部分是将 ∇T 的时间属性值与有 ∇T 参与的减操作的结果元组时间属性值进行合并;第 2 部分是插入 R 中与 ∇T 中相同的元组.

纯插入 ΔT 的更新传播计算式为

$$\left.\begin{aligned}\nabla^+\varepsilon_T &= (R-T)\propto_{OVERLAPS(\$1.T,\$2.T)\wedge(\$1.A_1=\$2.A_2)}\ \Delta T,\\ \Delta^+\varepsilon_T &= \nabla^+\varepsilon_T-\Delta T.\end{aligned}\right\} \tag{14}$$

(5) 选择操作 $\sigma_p(R)$:选择操作只是查找关系中的元组,没有选择完后的元组合并操作,而且 ∇R (ΔR)是基表 R 的纯删除(纯插入),不会对投影结果造成修改,所以有

$$\nabla^-\varepsilon_R=\sigma_p(\nabla R),\quad \Delta^-\varepsilon_R=\varnothing, \tag{15}$$

$$\nabla^+\varepsilon_R=\varnothing,\quad \Delta^+\varepsilon_R=\sigma_p(\Delta R). \tag{16}$$

(6) 连接操作 $R\triangleright\triangleleft_p T$:给定 R 的纯删除(纯插入):

$$\nabla^-\varepsilon_R=\nabla R\triangleright\triangleleft_p T,\quad \Delta^-\varepsilon_R=\varnothing, \tag{17}$$

$$\nabla^+\varepsilon_R=\varnothing,\quad \Delta^+\varepsilon_R=(\Delta R\triangleright\triangleleft_p T). \tag{18}$$

T 的更新传播算法与 R 相同.

2.2.3 时间前进

如果在视图定义中包含时间选择条件,即在选择条件中涉及时间属性或关键字 NOW,那么该视图内容就应该随着时间前进而不断更新,在 5 种基本历史关系操作中,只有选择操作和连接操作具备这种时间变化性质,则前面得到的式(5)~式(10)无须改变,而式(15)~式(18)要增加时间前进的作用成分.

(1) 选择操作 $\sigma_p(R)$:

$$\left.\begin{aligned}\nabla^-\varepsilon'_R &= \nabla^-\varepsilon_{R(p\leftarrow_p\ \text{at}\ t_1)}\oplus\sigma_{((p\ \text{at}\ t_1)\wedge\neg(p\ \text{at}\ t_2))}R,\\ \Delta^-\varepsilon'_R &= \sigma_{((p\ \text{at}\ t_2)\wedge\neg(p\ \text{at}\ t_1))}(R-\nabla R).\end{aligned}\right\} \tag{15′}$$

$$\left.\begin{aligned}\nabla^+\varepsilon'_R &= \sigma_{((p \text{ at } t_1)\wedge\neg(p \text{ at } t_2))}R,\\ \Delta^+\varepsilon'_R &= \Delta^+\varepsilon_{R(p\leftarrow p \text{ at } t_2)} \oplus \sigma_{((p \text{ at } t_2)\wedge\neg(p \text{ at } t_1))}R.\end{aligned}\right\} \tag{16′}$$

上式中, $\nabla^-\varepsilon_{R(p\leftarrow p \text{ at } t_1)}$ 表示将式 $\nabla^-\varepsilon_R$ 中的条件 p 替换为 p at t_1,其余类似.下同.

(2) 连接操作 $R \rhd\lhd_p T$:

$$\left.\begin{aligned}\nabla^-\varepsilon'_R &= \nabla^-\varepsilon_{R(p\leftarrow p \text{ at } t_1)} \oplus R \rhd\lhd_{((p \text{ at } t_1)\wedge\neg(p \text{ at } t_2))} T,\\ \Delta^-\varepsilon'_R &= (R-\nabla R) \rhd\lhd_{(\neg(p \text{ at } t_1)\wedge(p \text{ at } t_2))} T.\end{aligned}\right\} \tag{17′}$$

$$\left.\begin{aligned}\nabla^+\varepsilon'_R &= R \rhd\lhd_{((p \text{ at } t_1)\wedge\neg(p \text{ at } t_2))} T,\\ \Delta^+\varepsilon'_R &= \Delta^+\varepsilon'_{R(p\leftarrow p \text{ at } t_2)} \oplus R \rhd\lhd_{(\neg(p \text{ at } t_1)\wedge(p \text{ at } t_2))} T.\end{aligned}\right\} \tag{18′}$$

由式(5)~式(14)和式(15′)~式(18′)就组成了 5 种基本历史关系代数操作的更新传播计算式.

3 结 论

时态视图是数据仓库的一个重要组成部分,因此时态视图的维护是数据仓库生命周期中的一项重要维护工作.本文针对历史关系操作的特点以及时态视图更新的两个来源,根据增量式维护准则,采用关系代数语言的表达形式给出了时态视图的增量式维护方法,用单个计算式就同时解决了基表更新和时间前进对时态视图的更新,而且采取了纯删除(纯插入)的增量式更新方法,不仅易于实现,而且大大节约了真正改写视图的时间.另外,文中采取的直接寻找法思想为其他历史关系模式下时态视图的增量式维护提供了一条途径.今后的研究目标是当定义时态视图的历史查询中同时涉及历史关系和传统关系时的维护算法.

References:

[1] Gupta, A., Mumick, I.S. Maintenance of materialized views: problems, techniques, and applications. IEEE Data Engineering, 1995, 18(2):3~18.

[2] Jagadish, H.V., Mumick, I.S., Silberschatz, A. View maintenance issues for the chronicle data model. In: Proceedings of the 1995 ACM Symposium on Principles of Database System. New York: ACM Press, 1995. 113~124.

[3] Jensen, C.S., Mark, L., Roussopoulos, N. Incremental implementation model for relational databases with transaction time. IEEE Transactions on Knowledge and Data Engineering, 1991,3(4):461~473.

[4] Jensen, C.S., Mark, L. Differential query processing in transaction-time databases. In: Tansel, B., ed. Temporal Databases: Theory, Design, and Implementation. London: Benjamin/Cummings, 1993. 457~491.

[5] Bakgaard, L., Mark, L. Incremental computation of time-varying query expressions. IEEE Transactions on Knowledge and Data Engineering, 1995,7(4):583~589.

[6] Yang, J., Widom, J. Maintaining temporal views over non-temporal information sources for data warehousing. In: Schek, H-J., Saltor, F., Ramos, I., eds. Proceedings of the 6th International Conference on Extending Database Technology Berlin. Germany: Springer-Verlag, 1998. 389~403.

[7] Allen, J.F. Maintaining knowledge about temporal intervals. Communications of the ACM, 1983,26(11):832~843.

Maintaining Temporal Views in Data Warehouses*

LI Qi[1], BAI Ying-cai[1]

[1](*Shanghai Telecom Billing Center*, *Shanghai* 200071, *China*);

[2](*Department of Computer Science and Engineering*, *Shanghai Jiaotong University*, *Shanghai* 200030, *China*)

E-mail: happlq@263.net

http://www.sjtu.edu.cn

Abstract: An important use of data warehouses is to provide historical information by temporal views. Because it makes temporal extension to traditional relation data model, the materialized temporal views may be needed updated not only when base relations change, but also as time advances. So presently the research results for the non-temporal views are not applicable for the temporal views, and also the existed temporal view maintaining techniques are not appropriate for the data warehouses. Based on a historical relational model, according to the pure deletion and pure insertion concept, the change propagation are presented in this paper computation in relational algebra format for the five primitive historical relational operators. The incremental computation of temporal views defined by the five operators can be obtained by iterative method. At the same time, the pure insertion and the pure deletion concept accepted by this paper can be used in the temporal view maintenance under other historical relational model.

Key words: data warehouse; temporal operation; view maintenance; historical relational schema; incremental computation

⁎ Received September 28, 2000; accepted March 12, 2001

※※※

中国计算机学会系统软件专业委员会 2002 年学术年会

征文通知

中国计算机学会系统软件专业委员会定于 2002 年 10 月在杭州召开中国计算机学会系统软件专业委员会 2002 年学术年会。会议将由南京大学计算机软件新技术国家重点实验室主办，浙江省计算机学会协办。

一、征文范围(征文范围以下列专题为主，但又不限于此)

软件自动化与形式化	操作系统
软件语言及其处理	分布对象技术和软件中间件
嵌入式系统	软件安全性
对象与软件 agent	分布与并行处理

二、征文要求

论文应未曾在国内外杂志上或会议上发表过。字数一般不超过 6000。

论文应包括题目、作者姓名、单位、通讯地址、邮编、Email 地址、中英文摘要、关键词、正文、参考文献以及文章联系人与主要联系方式。

论文一律为 A4 打印稿，一式三份，用 Word 排版。同时也接受电子投稿，电子稿只接受 PDF 文件。

征文请寄： 南京大学计算机软件新技术国家重点实验室 陶先平（210093）

电子稿投稿：txp@softlab.nju.edu.cn

三、关键日期

截稿日期：2002 年 8 月 10 日　　　录用通知日期：2002 年 9 月 15 日

四、联系方式

陶先平，025－3593694（O），025－3593670（Fax）

txp@softlab.nju.edu.cn