数据仓库实化视图的联机维护

刘小宁 马光志（华中科技大学计算机系 430074）

摘要：数据仓库实化视图的联机维护是数据仓库系统维护的一项关键技术，采用这种技术，能够在不影响用户正常业务的情况下，实现数据仓库的实化视图数据的及时更新。联机分析处理(OLAP)作为数据仓库的一个主要应用，在数据仓库实化视图的联机维护的过程中会面临严重的数据不一致问题。为了解决这个问题，本文引入'维护库'(Maintaining Database)的概念，提出基于事务触发的视图维护算法TVM，通过消息应答机制实现实化视图与数据源的数据一致性。

关键词：实化视图 实化视图在线维护 监视器 集成器 维护库 TVM

1 引言

数据仓库中的信息是由各个独立分布的数据源的数据汇集而成。当数据源的原始数据发生变化时，数据仓库的维护系统就要将这些变化反映到数据仓库中，使数据仓库中的数据得到及时的更新。现有的数据仓库产品大都采用脱机维护方式，即用户在数据仓库不进行业务处理时维护。随着全球业务的发展、24小时时间的限制、实时系统的要求，使得脱机维护方式越来越难以满足需要。因此，必须发展数据仓库的联机维护技术，以保证在不影响业务处理的条件下及时更新数据。

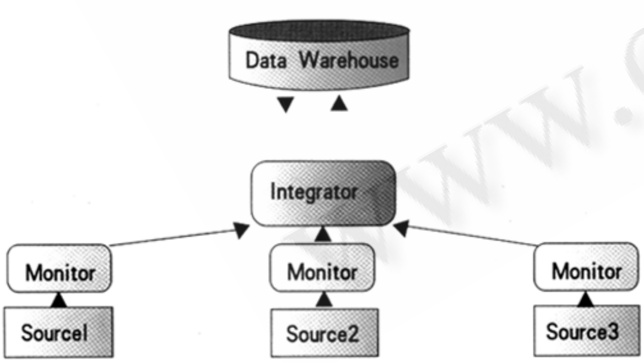

图 1 数据仓库体系结构

在数据仓库的联机维护中，维护数据仓库的实化视图是一项关键的技术。对于数据仓库联机分析处理(OLAP)的一些操作，比如下钻查询，数据仓库不仅要查询实化视图，而且要访问数据源，才能得到正确的查询结果。如果数据源中的数据已被修改，但是并未更新到实化视图中去，那么下钻查询就会产生不正确结果。

Stanford大学提出了数据仓库体系结构模型WHIP[1](如图1所示)，并基于快照方式提出了实化视图的维护算法，解决了非实时系统数据不一致问题。

在图1所示的数据仓库系统中，实化视图的维护主要由集成器(Integrator)和监视器(Monitor)完成。各模块之间相互传递消息，通过消息来协调各模块的操作，保证数据的一致性。

2 TVM算法

在Stanford大学提出的数据仓库维护算法中，数据源的数据变化信息通过快照方式[1]传递到数据仓库，对于实时性要求不高的领域，快照方式一般可以满足需要，但在实时性要求较高的系统中就会出现数据的时效问题。

本文采取了在数据源端加载触发器的消息传递方法。数据源端一旦进行增、删、改操作，触发器就被触发，通过监视器(Monitor)将数据变动传给集成器(Integrator)。集成器(Integrator)中加载了一个称为“集成维护库(简称维护库)”的数据库，管理数据仓库中视图与源数据的数据一致性问题。维护库的建立不仅能保证在联机维护状态下用户查询的一致性，还能减少系统开销，解决更新异常的问题。本文提出的TVM算法采用应答机制和脉冲算法[2]，在保证在对数据仓库中实化视图联机维护的同时，使OLAP查询得到同数据源一致的结果。

为简化起见，以下只考虑一个数据源的情况，多个数

据源算法与此类似。

2.1 消息传递类型

在数据仓库视图维护过程中，通过消息传递实现监视器、集成器以及数据仓库视图管理机制的协同工作：监视器监测源数据的变化，并将变化信息传递给集成器。集成器接受这些信息，经过集成汇总后传入数据仓库，通过数据仓库中的视图管理机制对视图更新，保持视图与数据源数据一致。这些消息主要包括数据源发出的数据变化通知消息、数据仓库端发出的结果应答消息和集成器(Integrator)发送的维护通知消息。

(1) DB端发出的数据变化通知消息db_update，共有三种消息：

insert(r,t) 在关系r中插入元组t

delete(r,t) 在关系r中删除元组t

modify(r,t1,t2) 关系r中用元组t2更新元组t1

(2) DW端视图维护返回的结果应答消息dw_update：视图维护已完成(TRUE)，未完成(FALSE)

(3) 集成器(Integrator)端发送的消息dw_link，通知数据仓库的视图管理机制对视图进行维护。

2.2 维护库的数据表示

维护库是本算法中用于进行数据一致性维护的数据库，它被加载在集成器(Integrator)中，主要用于保证数据仓库中视图与源数据的数据一致性。设维护库的元组结构为T_Construct，元组结构如图2所示：

V-name	R-name	T_turple	Flag	Vflag

图 2 元组结构图

其中：V_name：实化视图名

R_name：关系名

T_turple：元组

Flag：标志位，有三个值：

I：表示此记录的元组是刚插入的；

D：表示此记录的元组已被逻辑删除；

M：表示此记录的元组刚被修改(修改前的元组作为旧版本被移入cache中或缓冲池中)。

Vflag：标志数据仓库实化视图更新状态：

TRUE：已更新；

FALSE：未更新。

R_name是指数据源中与实化视图相关的字段组成的关系，数据源中其他与实化视图不相关的字段不予考虑。

定义：设DR为数据源中关系R(DR)的属性集，DV为数据仓库实化视图V(DV)的属性集，若存在一属性集D，D⊂ DR，且D⊂ DV，则由属性集D组成的关系RD(D)称为视图-关系相关项，V(DV)称为关系RD(D)的关系-视图相关项。

对于关系T_construct中的每条记录，字段R_name的属性值是字段V_name属性值的视图-关系相关项。每个实化视图中的数据都建立在一个或几个数据源表的基础上，而一个关系中的某个元组更改，会影响多个视图进行更新，所以V_name与R_name之间是多对多的关系。

2.3 实化视图的更新

若在数据源端的关系R中对元组t做了一次更新操作Mod，则监视器(Monitor)获取此操作信息，将消息传递给集成器(Integrator)，由集成器(Integrator)对维护库进行维护操作，并通知相关视图进行更新处理。

若关系R存在n个关系-视图相关项V_1，V_2，…V_n，那么关系R中一个元组的更新会影响这n个视图更新。集成器需要发n条消息通知数据仓库视图管理机制对视图进行维护，并在关系T_Construct中插入n条记录：

for (i=1;i ≤ n; i++)

{Insert into T_Construct (V_name,R_nameT_turple, Flag,Vflag)values(Vi,R,t,Mod,False);}

对于任意数据仓库实化视图Vi(1<i<n)，更新结束就置相应记录的Vflag属性为True。只有关系R的所有关系-视图相关项V_1，V_2，…V_n对应记录的Vflag-True都成立，数据仓库的这一次更新操作才全部完成。

由于数据仓库业务处理的需要，在更新并未完全结束时，数据仓库中仍可进行的查询和向下钻取操作，但有一定的限制条件：在关系T_Construct中，属性Vflag为True的记录对应的视图已更新完毕，可支持决策分析并进行所有的OLAP操作。对于属性Vflag为False的记录，Flag有三种可能的值：标志位为I对应的元组，更新变化还未反映到实化视图中去，不可能参加OLAP查询运算；标志位为D对应的元组，该元组只被逻辑删除，可以参加OLAP操作和向下钻取；对于标志为M对应的元组，可以进行OLAP和向下钻取查询操作，但是参与操作的数据是被移入cache中的修改前元组的值。

2.4 TVM算法

(1) 监视器(Monitor)在数据源中的与实化视图相关的字段上加载了触发器。当相关字段发生更新操作(增、

删、改)，触发器就被触发，通过监视器(Monitor)，将消息 db_update 传送到集成器(Integrator)中。

(2) 在 Integrator 中的维护库中间向 T_Construct 插入 n 条记录，以便对 n 个受此更新操作影响的实化视图进行更新控制。并采用脉冲算法 [2]，解决更新异常问题。

(3) 集成器(Integrator)发送消息 dw_link，通知了数据仓库进行实化视图的维护，将视图中相对应的项更新。

(4) 视图更新后，数据仓库端向集成器(Integrator)发送消息 dw_update，通知集成器(Integrator)此视图的更新操作已经完成。

(5) 集成器(Integrator)接受到 dw_update 的消息后，将 T_Construct 中相对应的记录进行更新，将 Vflag 项置为 true。

重复第(4)、(5) 步，直到 n 个实化视图都更新完毕。

3 TVM 算法特点

本文提出的TVM算法不仅能保证数据仓库中视图的联机维护，也保证了向下钻取得到的细节数据和宏观数据的一致性。该算法具有以下特点：

(1) 用户可以随着使用数据仓库。数据仓库不再有专门的对外服务时间和更新时间，这两者可以同时进行，极大的提高了系统的可用性；

(2) 数据仓库中的多个实化视图之间，以及它们与数据源之间的数据始终是一致的，因此支持下钻查询，且返回给用户的查询结果是一致的；

(3) 基本保持现在的数据库系统不变；

(4) 解决更新异常问题，保证联机维护结果的正确性；

(5) 适用于实时性较高的系统。

4 结束语

随着数据仓库的应用日益广泛，商务数据分析的即时性增强，实时数据仓库的联机维护将成为必然的发展趋势。本文提出的TVM算法能够满足实时性要求较高的系统，解决了数据仓库联机维护数据的一致性问题和更新异常问题。采用该算法的系统将能更准确地提供决策信息。

参考文献

1 Joachim H. The Stanford Data Warehouse Project Stanford University 1995.

2 Zhu-ge Y. Consistency Algorithms for Multi-Source Warehouse View Maintenance Stanford University 1997.