流水线处理技术在数据集成中的应用

房元平,许娇阳,葛 珂
(暨南大学 计算机科学系,广东 广州 510632)

摘 要:在数据集成中,经常碰到大数据量的集成问题,基于数据仓库方式的数据集成技术是一种比较流行的集成模式,提高该集成模式的查询以及实化视图的初始化效率、响应速度,并防止内存溢出,是数据集成中非常关注的地方。在基于数据仓库方式的数据集成模式中,利用基于内存控制的流水线处理方法,提高查询以及实化视图的初始化效率。实验证明,以上方法较同步方法不仅提高了数据集成效率,而且实现了内存控制。

关键词:数据集成;查询效率;内存控制;流水线
中图分类号:TP311.13 **文献标识码**:A **文章编号**:1674-7720(2010)24-0067-03

Application of pipeline processing technique in data integration

FANG Yuan Ping, XU Jiao Yang, GE Ke
(Department of Computer Science, Jinan University, Guangzhou 510632, China)

Abstract: In the data integration field, often encounter a large amount of data, an approach based on data warehouse data integration technology is a popular integration model, how to improve query and the initialization of materialized view's performance, response speed, and to do memory control to prevent memory overflow at this mode, are very concerned in data integration. In this paper, based on data warehouse data integration model, used a pipeline processing method to improve the efficiency of query initialization of materialized view, experiments and theory show that the above method not only improve the efficiency of data integration, and achieved a good memory control.

Key words: data integration; query efficiency; memory control; pipeline

随着个人计算机和计算机网络的飞速发展,以及信息化的高速推进,互联网提供的信息总量也在迅猛增长。如果企业和社会组织实现数据共享,可以使更多的人更充分地利用已有的数据资源。可是为不同应用服务的信息都存储在许多不同的数据源之中,数据内容以及数据格式千差万别,且其管理系统也各不相同。如何对这些数据进行有效的集成管理,屏蔽这些信息的异构,并提供一个统一的访问接口以透明地访问各信息源,成为一些大型企业或社会组织关心的事情。数据集成正是在这一背景下提出的。

1 基于数据复制方法的集成模式

数据复制方法[1]是当前比较常用的数据集成模式,该方法将各个数据源的数据复制到与其相关的其他数据源上,并维护数据源整体上的数据一致性、提高信息共享利用的效率。这种方式可以复制信息源的整个数据,也可以是信息源的部分信息。数据复制方法在用户使用某个数据源之前,将用户可能用到的其他数据源的数据预先复制过来,如果用户要使用的数据已经被复制,则只需要查询该集成信息源,并与中介器/包装器的虚拟数据集成[2]相比,大大提高了系统处理用户请求的效率。

基于数据复制方式最常见的一种方法是数据仓库方法[1]。该方法将各个数据的全部或者部分数据复制到数据仓库,用户像访问普通数据库一样直接访问数据仓库。该方式实现了对物理数据库语义异构的屏蔽和数据访问的控制,提供了一个统一的数据逻辑视图来隐藏底层的数据细节。图1所示为一个典型的数据仓库体系结构图[3]。

在该集成模型中,每一个数据源对应一个监视器(Monitor),监视器负责收集所需要集成的信息源中数据的变化以便上报给集成端(收集的方式有如下类别:针对信息源有日志的情况,可以通过日志分析提取要上报的

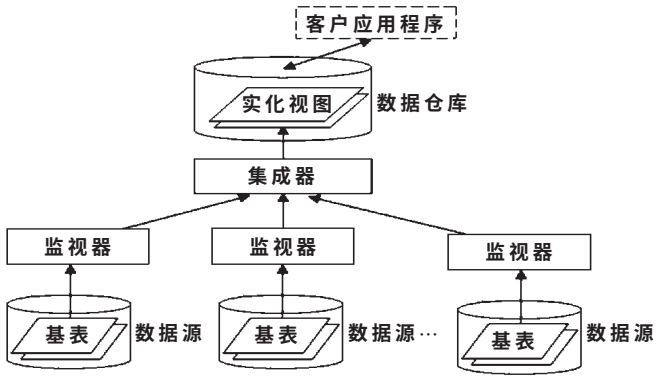

图 1 数据仓库体系结构

增量;对于没有日志情况可以通过触发器方式或者快照差分方式获取信息源的增量),同时监视器还具有一个包装器的功能,提供信息源的数据查询提取以及类型转化功能。当作为数据查询功能的时候,不仅将数据初始化同步到数据仓库中,同时也相当于一个服务器,不断侦听来自于集成器的命令查询请求,当有请求到达时,执行查询,并将该监视器对应的数据源的数据包装成基于公共类型的数据,或以 XML 文件的方式和固定大小对象数据块的方式传递给集成器,然后集成器负责将提取后的数据进行合成。其中监视器与集成器中的通信流程如图 2 所示。

图 2 监视器与集成器通信流程图

2 基于内存控制的流水线处理方法

从上面的数据仓库体系结构可知,监视器必须具备一个包装器(wrapper)的功能。数据仓库端保存的数据是各底层信息源的部分副本(一般情况为访问非常频繁),但是不是很频繁的访问数据还是保持在底层信息源端,当一个 OLAP 查询(如下钻)经过查询分解后,不能在数据仓库端获取,而必须通过一个甚至几个底层信息源端的查询,然后在集成器端进行结果的合并(如要通过两个底层数据库中表的连接操作)才能获取查询结果。在实化视图初始化时,提高查询的效率以及提高实化视图初始化的时间,是非常重要的。

本文关注的便是如何提高查询效率、响应速度、集成端的处理效率,以及在提交查询后,如何减少集成端的空闲等待时间,并且在大数据量的情况下同时做到内存控制,以防止在大数据量的情况下查询导致内存溢出。

在解决提高查询效率、响应速度、集成端的插入效率的同时,防止内存溢出以及在减少集成端的空闲等待时间方面,利用了基于生产者/消费者的流水线处理方法,该方式主要思想是实现服务器与客户端的流水并行[4],查询的结果以固定大小数据块的形式分块组装,并在监视器端与集成端都使用一定大小的缓冲队列来暂存这些数据块,以有效防止内存溢出。以一次实化视图的初始化过程为例,描述该方式的算法流程为:

(1)集成端发送带全局查询 QID(该查询 QID 为全局唯一的,通过客户端 API 自动生成)的 SQL 查询命令(结果查询重写),并通过通信平台将该查询命令放入服务器端执行队列中,同时预设一个数据块计数为 sum(该计数为服务器端初始要发送的数据块个数),然后集成端监听接收队列;

(2)监视器端从命令队列中取出查询命令,创建查询管理器(Data Query Manager),并将该查询管理器与查询 QID 作为一个键值对放入进程全局的哈希表(Concurrent Hash Map)中,然后通过该查询管理器中的 excuteQurey()方法启动查询线程,该查询线程将获得的记录组织成数据块(Data Object Block),放入固定大小的数据块缓冲队列中,并在该队列满时,查询线程暂停,不满时继续查询,直到最后一块为止。同时启动发送固定大小的数据块的线程,该发送线程从缓冲队列中取出数据块,发送给客户端,直到发送的最后一块,该发送线程终止;

(3)当有数据块到达客户端的数据块接收队列时,判断该块是否为最后一块,如果是,则设置所有块是否到达的标志"flag=true",并通知客户端进行处理,客户端处理线程从队列中取出一个数据块进行处理(对实化视图初始化,处理的方式就是将该数据块插入到数据仓库的实化视图中),并将数据块计数 n 减 1,再判断该数据块计数是否小于客户端要缓冲的个数 N,并同时判断 flag 的值,如果 sum<N,且 flag= =false,则发送从服务器端调取固定数目 K 数据块的命令(该命令带 QID,以便到服务器端时找到之前的查询管理器),同时设置 sum=sum+K;

(4)服务器端接收到客户端的数据块调取命令,分离出里面的 QID,从进程全局的哈希表中找到与该 QID 对应的查询管理器,并调用里面的发送固定数据块的方法以启动发送固定数目数据块的线程,该线程与步骤(2)中发送线程相同;

(5)重复步骤(3)、步骤(4),直到查询的最后一块到达客户端,与此同时,服务器端的查询管理器也从全局的哈希表中移除。

3 性能测试与分析

与流水线处理方法相对应的一种方法为同步方法,即通过查询先将底层信息源的结果组装在一起,一次传给集成端处理。由于采用的都是对象数据块的形式,因此用于与流水线对比的同步方法的算法思想为:

(1)客户端发送带全局查询 QID(该查询 QID 为全局

唯一的,通过客户端 API 自动生成)的 SQL 查询命令(结果查询重写),并通过通信平台将该查询命令放入服务器端执行队列中;

(2)服务器端接收到查询命令,执行查询,将所得的结果存放于文件中,然后一次发送给客户端;

(3)客户端接收到关于本次查询结果集的文件,然后处理该结果集文件。

将基于内存控制的流水线处理方法与同步方法在以下实验环境下进行测试对比,为减少误差,多次测试得出平均值,有如下数据:

监视器端与集成端采用相同配置环境,相关配置为:

CPU:Intel(R) Core(TM)2 Duo CPU E4500 @ 2.2 GHz;操作系统:Windows XP;内存:2.0 GB;数据库:Oracle 9i;JDK 版本:1.6.0_07;开发工具:Myeclipse6.5。

本实验性能测试如图 3 所示,可以看出,与传统的同步方法相比,采用本文算法具有较好的性能特性,主要在于基于内存控制的流水线处理过程是一个监视器端与集成器端并行流水线运行的过程,并充分应用了现在多处理器多线程处理的技术,减少了集成端空闲等待的时间。

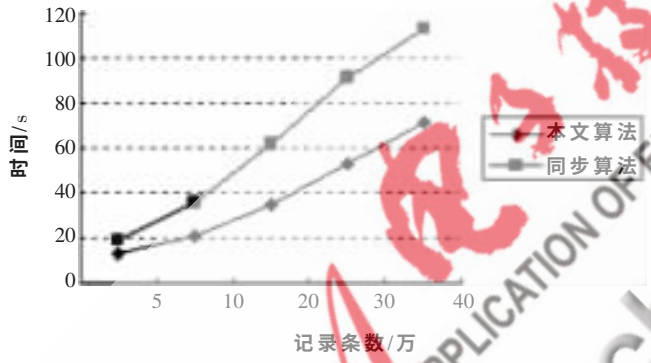

图 3 性能分析

设查询信息源并包装所有数据成公共类型数据块的时间代价为 Cost(Q),传输放入文件中的所有数据块到集成端的时间为 Cost(T),集成端将传输过来的数据解析并初始化到数据仓库的时间为 Cost(P),则基于同步方法的时间代价为:Cost(Q)+Cost(T)+Cost(P)。

设查询信息源并包装查询的数据成公共类型数据块为一块的时间代价为:Cost (Q_1),传输其中一块数据块到集成端的时间为 Cost(T_1),集成端将传输过来的一块数据块解析并初始化到数据仓库的时间为 Cost(P_1),因为这里数据块是个固定的常数,则基于本文的算法的时间代价为:Cost(Q_1)+Cost(T_1)+Cost(P_1)+max(Cost(Q)−Cost(Q_1), Cost(T)−Cost(T_1), Cost(P)−Cost(P_1)),其中 max 为各处理逻辑减去初始处理的最大时间。

从上面理论上可以分析得出,基于内存控制的流水线处理技术较同步技术可以更好地提高效率。同时还存在几个问题:

(1)当集成端需要 OLAP 查询或实化视图初始化比较多时,仍然会出现内存溢出的问题,这时可以应用线程池技术[4],有效控制这类线程运行的数量,同样,监视器端也使用这种方案。

(2)当集成端与监视器端进行流水线处理时,如果监视器端与集成端出现网络中断,或者其中一个出现突发事件(如断电)时,之前的一些过程就需要重做,并回滚。特别是针对网络中断的情况,容易造成监视器端查询线程的线程泄漏,即集成端认为之前的操作没成功,然后重新进行操作,然而监视器端的处理线程却还没完。避免这些情况出现的解决方案为:设置一个超时,当达到设定时间而这一流水处理过程未进行时,自动中断这些处理流程,或者可以在监视器端对查询组装后数据块分块存储在硬盘上,然后进行文件数据块的发送,这样减少了块之间的命令的交互逻辑,而且有效地控制了线程泄漏,但是也增加了文件的读写与控制,增加了 I/O 开销。

数据集成仍然是一个比较热门的研究点,在基于数据仓库方法的数据集成中,分析了实化视图初始化以及 OLAP 查询中面对大数据量处理的问题,应用了基于内存控制的流水线处理方法,充分利用了 Java 的多线程处理技术,并从实验和理论上分析了该方法较同步方法的优点。

参考文献

[1] LABIO W J, GARCIA-MOLINA. Efficient snapshot differential algorithms for data warehousing[C]. In Proceedings of VLDB Conference, Bombay, India, September 1996: 63-74.

[2] 陈跃国,王京春.数据集成综述[J].计算机科学,2004,31(5):48-51.

[3] 何震瀛,李建中,高宏.Web 数据仓库异步迭代查询处理方法[J].软件学报,2002,13(2):214-218.

[4] GOETZ B.JAVA 并发编程实践[M].方妙,韩锴,译.北京:电子工业出版社,2007.

[5] MOSTAFA E,MUSTAPHA B,LOUBNA C.An advanced XML mediator for heterogeneous information systems based on application domain specification. International Journal of Computer Science & Applications[J].2006,3(2).

(收稿日期:2010-07-23)

作者简介:

房元平,男,1983年生,硕士研究生,主要研究方向:数据库。

许娇阳,女,1986 年生,硕士研究生,主要研究方向:数据库、信息集成。

葛珂,男,1985 年生,硕士研究生,主要研究方向:数据库、信息集成。