可扩展数据仓库若干关键问题研究与分析

秦学勇[1],姚燕生[2]
(1.安徽建筑工业学院 计算机与信息工程系,安徽 合肥 230022;
2.中国科学技术大学 自动化系,安徽 合肥 230022)

摘 要:当大量复杂异构的数据不断进入数据仓库以后,最初的配置渐渐适应不了新的环境,因此数据仓库必须要设计成为可扩展的体系结构,从而能够满足将来数据仓库的性能要求,它能够随着数据、最终用户、复杂性和功能性的增长而增长,并且在增长的同时其性能水平并不会下降。文中通过对数据集成、高效查询和非规格化等几个可扩展数据仓库关键问题进行研究,使得当大量异构数据涌入可扩展数据仓库中时系统性能不会下降,很好地满足决策支持,取得较好效果。

关键词:可扩展;数据仓库;数据清洗;位图;非规格化
中图分类号:TP311.138 **文献标识码**:A **文章编号**:1673-629X(2006)12-0136-03

Research and Analysis on Some Key Problems of Scalable Data Warehouse

QIN Xue-yong[1], YAO Yan-sheng[2]
(1. Dept. of Computer and Info. Eng. of Anhui Inst. of Architecture and Industry, Hefei 230022, China;
2. Automatization Dept. of Univ. of Sci. and Techn. of China, Hefei 230022, China)

Abstract: Data warehouse should be designed with scalable architecture so that it can satisfy the need of future applications. When mass data and more users entering into the scalable data warehouse, the system can finish tasks well and maintain a good performance. Through researching and analysis on some key problems of scalable data warehouse, we can get a good effect on system performance when a great deal of heterogeneous data swarming into it. Data integration, effective index and non-normalization are used to maintain the performance of system, and satisfy the need of decision support.

Key words: scalable; data warehouse; data cleaning; bitmap; nor-normalization

1 可扩展数据仓库

随着信息技术和数据库技术的飞速发展,数据处理的重点从传统的业务过程扩展到对业务数据的联机分析处理,并从中得到各种决策支持信息。数据仓库[1]技术为联机分析处理和决策支持提供了一个良好的技术解决方案,是基于大规模数据库的决策系统环境的核心。数据仓库不是一日建成的,而是通过不断的迭代方式来建立它,当大量的数据开始在数据仓库中堆积时,数据仓库的性能就开始成为一个重要的问题。所以数据仓库必须设计成为一种可扩展的结构,以便适应将来的数据环境。通过并行技术使得在数据仓库不断扩展时,其性能不会直线下降,而是保持良好的工作环境;而通过数据集成、高效查询和非规格化等方法可在不增加硬件的前提下扩展数据仓库性能。目前,国内外公司、团体和个人对数据仓库性能研究方面作了不少的工作,取得了不少成果,但很多问题也没有得到很好地解决。

收稿日期:2006-04-04
作者简介:秦学勇(1974-),男,安徽合肥人,硕士,讲师,研究方向为数据库与多媒体软件。

2 数据集成

数据集成[2]是数据仓库工程中很重要的部分,是将多个数据源中的数据结合起来存放在一个一致的数据存储之中,这里就是指数据仓库。对于现实世界中的同一实体,来自不同数据源的属性值可能不同,这是由于不同的操作型环境的设计人员对实体的不同的表示方法,实际应用中,在编码、命名习惯、实际属性、属性度量等方面没有什么是一致的,在各个不同的应用问题中,设计人员自由地做出他们自己的设计决策。标准的数据仓库结构中由很多部分组成,但是抽取数据、转换数据和装载数据进入数据仓库是最重要的部分。其实这四者有时关系区分并不是十分明确,存在着相互的包含关系,由于转换过程和数据清理是密不可分,故转换应该包括数据的清理过程[3]。成功的、高效的数据仓库解决方案很大程度上依赖于高效的抽取、转换和装载(ETL)部分。在大多数数据仓库解决方案中,在将数据装入数据仓库之前,要花费将近70%的时间和精力去集成大量的异构的数据流。

由于多种原因,在数据仓库中存在脏数据(Dirty Data)是一件很正常的事情,既然不能保证脏数据绝对不进

入数据仓库平台,那么就应该想办法去处理这些脏数据。数据转换过程是和数据清理分不开的,转换应该包括数据的清理过程,在这儿主要讨论发生在数据集成时刻的数据清理[4,5]。数据清理包括确认数据的正确性,校正不正确的数据,然后以有效格式转换为正确数据。数据清理是一个减少错误和不一致性、解决对象识别的过程。数据清理原理:利用有关技术如数理统计、数据挖掘或预定义的清理规则将脏数据转化为满足数据质量要求的数据。数据清理过程如图 1 所示。

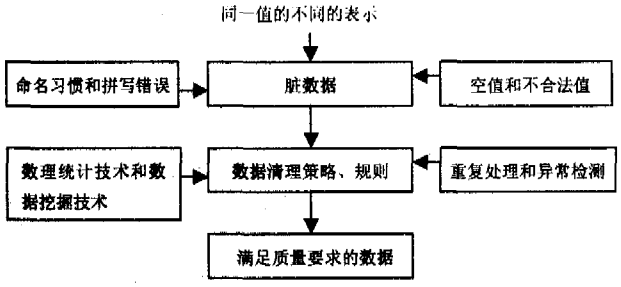

图 1　数据清理原理

在脏数据清洗算法上,一些研究机构提出了脏数据的预处理、排序邻居方法、优先排队算法、多次遍历数据清理方法、增量数据清理、采用领域知识进行清理、采用数据库管理系统的集成数据清理等算法。他们对于数据清洗算法的有效性的度量标准是,返回率(Recall):重复数据被正确识别的百分率;False - Positive Error:错误地作为重复数据的记录的百分比;精确度(Precision):算法识别出的重复记录中正确的重复记录的数据清洗研究百分比;计算公式:Precision = 100% - False - Positive Error。在数据清洗方面的研究大都是针对具体应用、领域开展的,同时由于数据清洗的领域相关性很强,通用的数据清洗可能受到很大限制。因此数据清洗框架的通用性很少有人关注。

3　高性能查询

如果你使用过大型数据库可能都有这样的体验:索引建立的好坏直接影响数据库的访问效率,有时一个简单的索引可能使同一程序的运行时间缩短几倍甚至十几倍。现在大多数的数据库系统用 B 树作为索引,该索引用一个树的序列结构和一个指向表中所有行的入口值,每个指针指向唯一的行号,行号的一般大小为几个字节。在数据仓库环境之中每个指针可能指向很多行,传统的 B 树索引并不能很有效地改善查询速度,这时考虑使用位图(Bitmap)索引[6]。对于只有少量的离散值来说使用 B 树索引不是很好,例如:某芯片公司生产五种不同档次的 CPU(A,B,C,D,E),在世界各地都有它的销售公司,如果你添加一个 B 树索引到产品列上可能没有什么用途,如果每种产品销售大致相等,每个索引项将指向 20% 的行。位图索引如表 1 所示,在建立位图索引时首先扫描整个表,创建一个位流,使每一位与表中某一单行的一个列值相对应,按所需索引列的分类列值,并根据在该列中找到的不同列值的数目决定建立该索引时需要多少个位图,然后建立那些位流以聚居所确定的索引。和 B 树索引一样,位图索引也是从一个原始表格分离出的结构,但是它们的结构大不相同。建立位图索引的第一步是确定索引列中离散值的准确数量,然后对于每个离散量按照以下原则来建立一个位图索引:如果某行包含一个特定值那么位图在此位置上包含一个 1 位,否则它包含一个 0 位。

表 1　产品类型的位图索引

A	B	C	D	E
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	1	0	0
0	1	0	0	0
0	0	0	1	0
0	0	0	0	1
1	0	0	0	0
0	0	0	0	1
1	0	0	0	0

当表格的列中包含少量的离散值时,位图索引比 B 树索引小得多。在这可就上述例子具体分析和计算:假设产品销售表有一千万行,并假设 B 树索引中的每个索引项有 10 个字节,那么一个 B 树索引将会需要 100MB(一千万索引项×每项 10 字节);而一个位图索引大约只需要 6.25MB((一千万行×每行 1 位×5 个位图)/每字节 8 位),它不仅节省存储而且读取速度要比 B 树索引快得多。通过逻辑 AND 和 OR 操作,位图索引也可以很容易地与相同表格中的其他位图索引组合在一起使用。例如:对上述例子来说,假如有 20 个不同位置的商店,再建立一个 Store_ID 的位图索引,通过对 A 位图和 #12 位图进行 AND 操作就可以计算出结果中 1 的数量,从而很容易知道商店 #12 销售了多少 A 类产品。对于某些查询位图索引特别快,如找出已经销售的 A 类产品的数量,数据库引擎只需要计算 A 位图中 1 的数量。由于每个位图都比较小且计算机在逻辑 AND 和 OR 方面是优越的,从而位图索引可以很好地提高查询性能。

4　非规格化

数据建模和规格化作用是产生一种完全没有数据冗余的设计方法,在规格化设计中仅有的冗余是因为外键关系和数据引用完整性的需要。偶尔在数据仓库的设计中引入冗余是有很大的意义的,非规格化[7]就是一种在关系数据库中引入数据冗余的处理方法。这种冗余的目的就是通过减少频繁的连接、聚集和推导来提高查询性能,这种技巧通过最佳化同一物理块中数据的物理存放位置、预导出和预计算频繁使用的属性来提高查询性能。数据仓库中可以使用的非规格化技巧主要有四种:创建数据阵列、预聚集数据、列复制和预连接表格。在数据仓库的探

索者看来非规格化可以实现其他一些有价值的目标,但是对信息的使用者来说非规格化并不能增加他们的性能需要,有经验的数据仓库管理员发现将经常一起使用的相关类型的数据存储在一起是一件非常有意义的事情。在图 2 中数据被规格化并且分开,这样每年中的所有月份被分别放置在不同的物理位置上,如果某个查询要查看一月、二月、三月等月份中的数据,那么系统必须到不同的地方去搜索数据。这里创建一个数据阵列,将相关的数据放在同一个物理位置上,现在如果做相关的查询只需要物理访问系统一次,因为这些数据已经被最佳地放置从而适应这样的要求。如果没有确认的和有规则的需求来同时查看一年中的某些月份,使用这种技巧并不能优化性能,信息使用者经常使用可预测和有规律的数据,所以这样的技巧可以很好地完成。图 2 所示为将相关类型数据一起存储的情形。

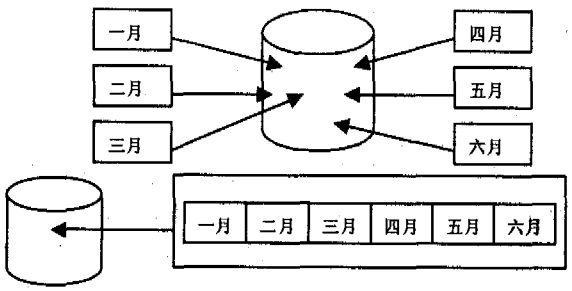

图 2 利用数据阵列并置相关数据

5 结束语

在使用并行技术从体系结构上扩展数据仓库的同时,通过对数据仓库几个关键问题的研究,使得当大量异构数据进入数据仓库时,数据仓库的性能不会变差,并维持良好的决策支持。

参考文献:

[1] Inmon W H. Building the Data Warehouse[M]. 王志海等译. 北京:机械工业出版社,2000.

[2] Strum J. Microsoft SQL Server7 数据仓库技术指南[M]. 刘汉宇等译. 北京:机械工业出版社,2000.

[3] Bergamaschi S, Castano S, Vincini M. Semantic Integration of Semistructured and Structured Data Sources[J]. SIGMOD Record, 1999, 28(1): 54 - 59.

[4] Bitton D, Dewitt D J. Duplicate Record Elimination in Large Data Files[J]. ACM Transactions on Database Systems, 1983, 8(2): 255 - 265.

[5] Kukich K. Techniques for Automatically Correcting Words in Text[J]. ACM Computing Surveys, 1992, 24(4): 377 - 439.

[6] Chan C Y, Ioannidis Y E. Bitmap Index Design and Evaluation [R]. Computer Sciences Dept., University of Wisconsin - Madison, 1997.

[7] Inmon W H, Rudin K, Buss C K, et al. 数据仓库管理[M]. 王天佑等译. 北京:电子工业出版社,2000.

(上接第 135 页)

验证措施来决定用户的访问权限,并对传输的数据进行加密,来保证数据在中间层和数据库层(底层)的安全传输。在诸如 Oracle, SQL Server 等大型数据库[5~7]提供了对象级(如视图、存储过程、表等)的甚至是记录行级的安全措施,Oracle8i 之后的版本支持 VPD(虚拟专用数据库)和细粒度存取控制(Fine Grain Access Control, 简称 FGAC)[6,7]。图 3 给出了 SQL Server 的数据安全策略。

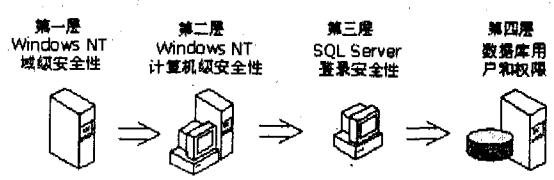

图 3 SQL Server 安全策略图

为了保证用户界面层和中间层之间的安全数据传输,同样采用用户身份认证的方式保证用户的合法性。并且在中间层的外面构筑企业级的防火墙,阻止非法入侵,保证用户的数据传输安全。

5 结束语

文中结合了工程实践,介绍了如何将微软.NET 框架与相关的大型数据库技术相结合,构建 ERP 中的库存管理系统。探讨其软件架构和相关的数据处理技术,并阐述了系统中采用的安全管理策略,为企业用户构建一个安全、方便、高效的应用系统,对于 ERP 系统中的其它部分实现同样具有一定的参考价值。

参考文献:

[1] 李嘉平. 大型 ERP 实施全接触[M]. 北京:电子工业出版社,2004.

[2] Knowles C, Mohr S. ASP.Net XML 高级编程——C# 编程篇[M]. 北京:清华大学出版社,2002.

[3] Kothari N, Datye V. ASP.NET 服务器控件与组件开发[M]. 邓春红,等译. 北京:机械工业出版社,2003.

[4] Silberschatz A, Korth H F. 数据库系统概念[M]. 北京:机械工业出版社,2003.

[5] 周立柱,冯建华,孟小峰,等. SQL Server 数据库原理——设计与实现[M]. 北京:清华大学出版社,2004.

[6] 胡欣杰. Oracle 9i 数据库管理员指南[M]. 北京:北京希望电子出版社,2002.

[7] Oracle Corporation. Oracle Db - security[EB/OL]. 2001 - 06. http://www.oracle.com/technology/deploy/security/oracle9i/pdf/9isecbpa.pdf.