第33卷 第12期 计 算 机 工 程 2007年6月
Vol.33 No.12 Computer Engineering June 2007
·软件技术与数据库· 文章编号：1000—3428(2007)12—0099—03 文献标识码：A 中图分类号：TP391

一种基于数据访问流的数据库索引优化方法

傅向华[1]，冯博琴[2]，王小民[1]，王志强[1]
(1. 深圳大学信息工程学院，深圳 518060；2. 西安交通大学计算机科学与技术系，西安 710049)

摘 要：提出了一种基于数据访问流进行索引优化的方法，该方法通过集成业务流程模型与数据模型，抽取了流程活动中包含的数据访问操作和流程的控制结构形成数据访问流，分析了数据访问流中数据访问操作的频度，为频繁数据访问操作所对应的数据项创建索引，达到改善数据访问性能的目的。实验结果表明，该方法提高了数据查询的速度。

关键词：索引优化；数据访问流；数据模型；流程模型

Index Optimization for Database Based on Data Access Flow

FU Xianghua[1], FENG Boqin[2], WANG Xiaoming[1], WANG Zhiqiang[1]
(1. College of Information Engineering, Shenzhen University, Shenzhen 518060;
2. Department of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049)

【Abstract】 A novel method called data-access-flow based index optimization (DAFIO) is proposed to provide automatic index select for database in order to improve the data access performance. By integrating the data model and process model of the IT systems and getting data access flow, the DAFIO method can analyze the data access flow to find those data item which are accessed frequently, and then create indexes for these frequent items. The experiment results show that the novel method is effective to improve the data access performance.

【Key words】 Index optimization; Data access flow; Data model; Process model

1 概述

数据模型是数据访问的基础，当信息系统的基本数据模型建立之后，需要进行必要的模型优化才能达到高效的数据访问性能。创建合适的索引和视图是提高数据访问性能的有效方法[1]。

目前，大多数的商业数据库系统(如DB2、SQL Server)在管理端都提供有数据访问优化器，数据库管理人员可以利用这些工具帮助创建索引和视图。一些研究人员也在不断对已有的索引和视图选择算法进行改进[2,3]，还有一些研究通过综合考虑数据库中的水平划分和垂直划分来改善数据库的访问性能，优化物理数据模型[4]。这些方法都基于对数据库中大量的静态数据和数据访问信息进行分析，需要大量的计算时间，具有很强的数据依赖性，很难用于规模较大的数据模型优化。

流程和数据是企业信息系统的2个重要组成部分。近年来基于流的技术越来越受关注，导致工作流引擎和基于流的编程模型的产生[5,6]。随着Web服务的兴起，流技术被扩展于Web服务的组合，用于对多个复杂、有状态交互的服务进行集成。商业公司纷纷提出了自己的Web服务组合语言规范，包括WSCI、BPML、XLANG、WSFL等以及最近由IBM、Microsoft、BEA、SAP、Siebel等公司联合提出的BPEL4WS规范[7]；与此同时，为方便用户进行设计和建模，一些工具提供商部分实现了流程模型与数据模型的集成，如CA公司的ERWin和BPWin、Sybase公司的PowerDesigner。

事实上，流程不仅是数据访问的基础，也是对数据访问的规范和约束，直接影响着数据的访问操作类型、访问频率等。因此，利用流程中有关数据访问的信息，同样可以对数据模型的优化起到积极的作用。

2 基于流程模型提取的数据访问流

2.1 业务流程模型

业务流程模型用于描述为完成某一特定业务目标所需要进行的一系列活动。业务流程由基本活动和结构化活动组成，其中，基本活动处理离散的细粒度的任务(如检索订单文档或更新客户记录)，结构化活动用于组合基本活动以解决更复杂的问题(如处理一个订购任务)，图1为在WebShpere Business Modeler5.0中建立的一个业务流程。

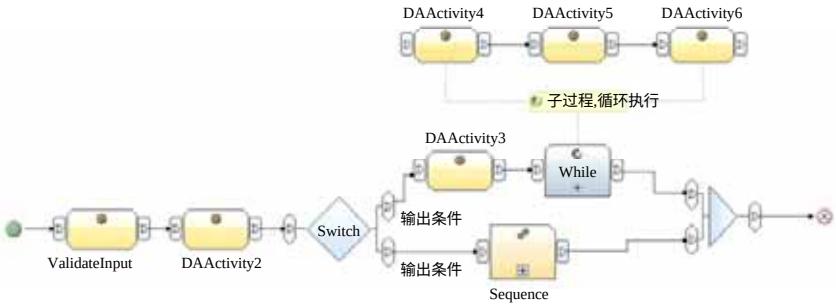


图1 业务流程的例子

基金项目：深圳大学科研启动基金资助项目(200648)
作者简介：傅向华(1977－)，男，博士、讲师，主研方向：软件工程，互联网信息检索；冯博琴，教授、博导；王小民、王志强，副教授
收稿日期：2006-08-02 **E-mail：**xianghuafu@gmail.com

该流程包括 ValidateInput、DAActivity1~6 等基本活动，也包含 switch、while、sequence 等结构化活动。BPEL4WS 作为广泛使用的流程描述语言，提供 2 种类型的流程表示方式：

(1)抽象流程。用于定义业务协议角色，可以提供给业务分析人员从抽象层面表示业务流程；

(2)可执行的流程。流程的逻辑和状态决定了在每个业务伙伴那里进行的服务交互的性质和顺序，从而决定了交互协议[7]。

2.2 数据访问流

流程通过调用多个功能实现一个全局的服务F，设流程P中的活动集合$A=\{a_1,a_2,...,a_n\}$所调用的功能集合对应为$\{f_1,f_2,...,f_n\}$，每个功能f_i的输入数据为$\{d_{i,k}^{(i)}\}_{k\in I}$，可能产生的输出数据为$\{d_{i,k}^{(o)}\}_{k\in I}$。当业务流程需要进行数据访问操作时，由于活动以消息的方式向相关的服务发送输入数据$\{d_{i,k}^{(i)}\}_{k\in I}$，请求数据访问功能$f_i$，接受从数据库返回的结果$\{d_{i,k}^{(o)}\}_{k\in I}$。因此，活动从业务的角度访问异构的数据源，其所请求的数据项与数据库之间存在对应关系。一个活动可能包含一个或多个数据访问操作。若从流程模型中抽取出所有的数据访问操作以及包含这些数据访问操作的流程控制结构，则可以得到一个数据访问序列，称为数据访问流。

数据访问流可用带权有向图$G=\{\{A,B\},E\}$表示，其中，A为所有的数据访问结点的集合；B为哑元{C，CE}和{P，PE}，分别表示选择的开始和结束、并行的开始和结束；E为数据访问结点之间的依赖关系。每个数据访问结点的权值根据该结点在业务流程中执行的次数，规定如下：并行、选择、序列活动为 1，重复活动为重复的次数。每条有向边的权值规定如下：序列、并行、重复控制结构中的边权值为 1，选择控制结构中的边权值为该边可能被执行的概率。另外，业务分析过程还会在业务流程中指定一些非功能性需求。因此数据访问流中的数据访问节点可以用该节点数据集合上发生的数据访问操作和非功能性要求表示，即$a_i=\{<d_{i,k},t_i,n_i,R>\}_{i\in I}$，其中，$t_i$为数据访问操作类型，$t_i\in\{select,delete,update,insert,null\}$；$n_i$为数据访问节点的执行次数；$R$为该节点的非功能性要求，如响应时间、吞吐量等。图 2 为一个数据访问流的例子。

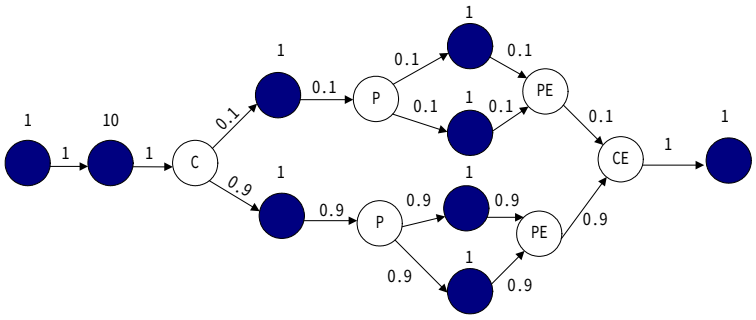


图 2 数据访问流的例子

3 基于数据访问流的索引优化

3.1 基于数据访问流的频繁活动分析

在数据访问流中，不同的数据项在整个流程中具有不同的频度，只有为那些被频繁访问的数据项创建索引才能达到改善数据访问性能的效果。而被频繁执行的活动所对应的数据项必然会被频繁地访问，因此，可以通过分析数据访问流的控制结构，确定频繁活动。

定义 1 (流程的频繁度FF) 若业务流程P直接调用或间接调用的子流程集合为$PS=\{P_r\}_{r\in I}$，P_r在整个流程中被调用的次数为N_r，则流程P_r的频繁度定义为$N_r/\sum_{r\in I}N_r$。

定义 2 (子流程的活动频繁度AFF) 若流程P_r中的任意数据访问活动节点$a_{r,k}$被调用的次数为$n_{r,k}$，则P_r的活动频繁度定义为$n_{r,k}/\sum_{k\in I}n_{r,k}$。

定义 3 (活动的频繁度 AF) 任意活动$a_{r,k}$的频繁度定义为该活动的访问次数与总活动访问次数的比率，则$n_{r,k}/\sum_{P_r\in PS}\sum_{k\in I}n_{r,k}$。

定义 4 (频繁查询活动) 活动的 AF 值大于关键查询活动频繁度阈值$MAXSearch_{\mathrm{AF}}$。

定义 5 (频繁更新活动) 活动的 AF 值大于关键更新活动频繁度阈值$MAXUpdate_{\mathrm{AF}}$。

可以采用多种图遍历算法来计算得到活动的频繁度，而且，可将常用的优化方法表示成规则，建立索引优化的规则库，根据这些规则确定出频繁度的阈值，从而确定出频繁查询活动和频繁更新活动。

3.2 基于数据访问流的索引优化

在索引选择过程中，为聚族索引和非聚族索引采用不同的策略。对于聚族索引的创建，关键是进行条件访问字列的统计。对于非聚簇索引的创建，需要权衡索引对查询速度的加快与降低修改速度之间的利弊。

假设数据访问节点$a_{r,k}$的频繁度为fre，该节点包含m个数据访问操作，每个操作对于某列的读写次数分别为R_s和W_s，则该节点中列的有效读数值为

$$Eff\_R_k=\sum_{s=1}^{m}fre*R_s$$

该访问列的有效写数值为

$$Eff\_W_k=\sum_{s=1}^{m}fre*W_s$$

对于整个数据访问流，某列的有效读写值为流程中所有访问节点的有效读写数值之和，分别为

$$Eff\_R=\sum Eff\_R_k \qquad Eff\_W=\sum Eff\_W_k$$

由于索引对数据的读访问有优化的作用，但又会影响写的速度。因此，用户在创建索引时可以设置 2 个权值：R_Ratio和W_Ratio。其中，R_Ratio表示读访问对索引的需求程度；W_Ratio表示写访问对索引的排斥程度。因此，列的加权分值为

$$WScore=Eff\_R*R\_Ratio-Eff\_W*W\_Ratio$$

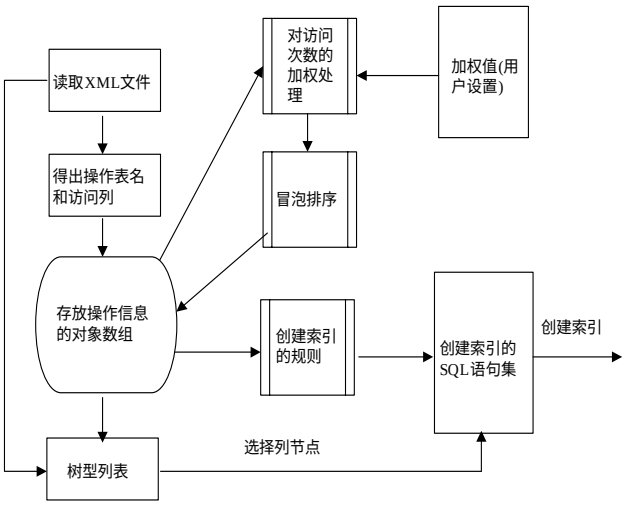
可以根据加权分值的大小来创建非聚族索引，加权分值越高的列，对索引的需求度越强。而对于条件访问列，其在访问过程中的作用是对该列的值进行给定条件的比较或筛选（如>、<、=、group by），只包含对列的读取，不存在读写加权的问题，因此在条件列的筛选过程中，只需考虑有效的访问数值。

4 系统实现与实验结果比较

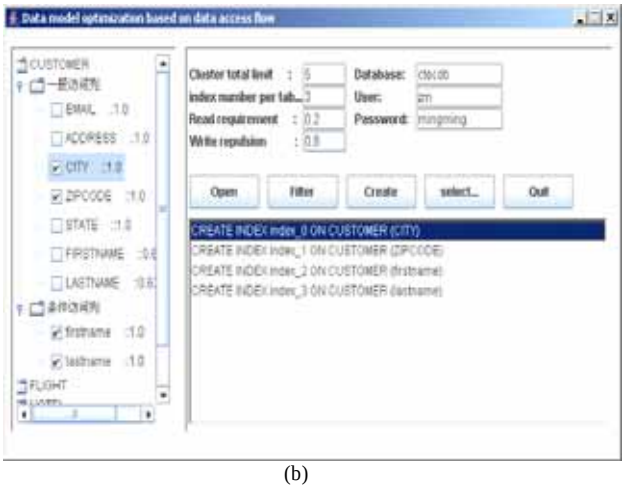
4.1 系统实现

基于 Eclipse3.0 集成开发环境和 JSDK1.5，笔者开发了

一套用于支持流程与数据集成的工具箱，包括流程数据关联分析器、数据访问构件生成器和数据访问优化器。开发人员只需指定流程中的业务项与数据模型中的表之间的关系、数据访问的要求，即可以自动生成合适粒度的数据访问构件，并通过对业务流程中的数据访问流进行分析，对频繁数据访问活动应用数据访问缓存模式、创建视图，为对频繁查询的数据列创建索引。其中，数据索引优化的处理流程如图 3(a)所示，首先分析流程数据关联模型中的数据访问操作类型、频繁度、数据表和数据访问的条件列等相关信息，然后对数据表中的列进行加权和归一化处理，得出每个条件列相应的分值，最后对该分值应用规则来创建索引；图 3 (b)为数据访问优化器的操作界面。



(a)



(b)

图 3 数据索引优化的处理流程和操作界面

4.2 实验结果比较

采用一个典型的旅行社预订业务Reservation应用场景[8]来说明基于数据访问流的索引优化方法。Reservation为旅行社为旅客提供旅馆、航班等预订业务的典型场景。首先将Reservation的流程模型文件和数据模型文件输入关联分析器，建立流程业务项与数据模型中表之间的映射关系，指定流程中数据访问活动的数据访问操作要求，自动生成平台无关的流程数据关联模型文件。然后，将该流程数据关联模型文件输入数据访问构件生成器，根据需要生成访问数据库的Java bean或EJB代码。并利用数据访问优化器对Reservation流程数据关联模型文件进行分析，自动创建视图和索引，利用缓存模式生成器自动将缓存模式应用到生成的数据访问构件上。

利用 Reservation 关联模型文件生成的数据访问构件，对创建索引和未创建索引的单表查询结果进行实验比较。随机在数据库表中添加 1 340 条记录，选择 20 个条件查询作为测试语句，如“ select * from zm.customer where firstname like 'm%' ”，对每条测试语句执行 500 次。实验结果表明，创建索引后的查询响应时间平均为 0.359s，未创建索引的平均查询响应时间为 0.719s。显然，索引创建对于查询性能的改善效果明显。在后续的工作中，将进一步考虑创建的索引对多表查询的影响。

5 结论

流程模型所包含的关于数据访问的执行顺序、性能指标等宏观信息非常有助于数据访问的性能优化。本文提出了一种利用全局性的数据访问流信息来改进数据模型的新方法，该方法通过建立数据模型和流程模型的关联关系，集成数据模型与流程模型，抽取既包含数据访问操作又包含全局性流程结构的数据访问流，然后自动分析数据访问流来确定所需选择的索引。实验结果表明，该方法可以达到比较好的优化性能。

参考文献

1 Caprara A, Fischetti M, Maio D. Exact and Approximate Algorithms for the Index Selection Problem in Physical Database Design[J]. IEEE Transactions on Knowledge and Data Engineering, 1995, 7(6): 955-967.

2 Agrawal S, Chaudhuri S, Narasayya V. Automated Selection of Materialized Views and Indexes for SQL Databases[C]//Proceedings of the 26th International Conference on Very Large Databases, Cairo, Egypt. 2000.

3 Chaudhuri S, Datar M, Narasayya V. Index Selection for Databases: A Hardness Study and a Principled Heuristic Solution[J]. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(11): 1313-1323.

4 Agrawal S, Narasayya V, Yang B. Integrating Vertical and Horizontal Partitioning into Automated Physical Database Design[C]// Proceedings of SIGMOD’04, Paris, France. 2004.

5 Leymann F, Roller D. Workflow-based Applications[J]. IBM Systems Journal, 1997, 36(1): 102-123.

6 Li Hongchen, Shi Meilin. Workflow Models and Their Formal Descriptions[J]. Chinese Journal of Computers, 2003, 26(11), 1456-1463.

7 Curbera F, Goland Y, Klein J, et al. Business Process Execution Language for Web Services[Z]. 2003. http://www.ibm.com/developerworks/library/ws-bpel/.

8 Keen M, Cavell J, Hill S, et al. BPEL4WS Business Processes with WebSphere Business Integration: Understanding, Modeling, Migrating[Z]. IBM RedBook SG24-6381-00, 2004.