第18卷 第1期
2008年1月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol. 18 No. 1
Jan. 2008

实用内存数据库核心及其在VLR系统中的应用

石英伟,姜 浩
(东南大学 计算机科学与工程学院,江苏 南京 210096)

摘 要:从实际应用的角度出发,引出内存数据库的定义及其系统特点,采用面向对象的软件设计方法设计内存数据库的内存组织结构,介绍内存数据库的核心对象并简单说明表对象操作方法的实现过程。结合移动网络中的VLR实体,简单阐述实用内存数据库在通讯领域的应用。

关键词:内存数据库;数据库核心;面向对象设计;访问位置寄存器
中图分类号:TP311.13 **文献标识码**:A **文章编号**:1673-629X(2008)01-0216-04

Core of Main Memory Database and Application in VLR System

SHI Ying-wei, JIANG Hao
(School of Computer Science & Engineering, Southeast Univ., Nanjing 210096, China)

Abstract: The definition of main memory database and its characteristics are proposed from an applied perspective. Employs object-oriented software design method to design the internal structure of the main memory database. And introduces its kernel objects and the table-object implementation process respectively. Finally, the application of main memory database in the field of communication is addressed together with the VLR entity of mobile network.

Key words: main memory database; database core; object-oriented design; VLR

0 引 言

访问位置寄存器(Visitor Location Register, VLR)是移动网络中的重要功能实体,存储和管理进入其控制区域内已登记的移动用户状态、位置更新、临时移动用户标识、移动用户漫游号码等相关信息。移动用户呼叫过程中,VLR为移动交换中心(Mobile Switching Center, MSC)提供必要的业务和数据支持,实现呼叫的接续。

VLR系统分为VLR移动应用(VLR Mobile Application Part, VLRMAP)与VLR数据库(VLR Database, VLRDB)两大部分。VLRMAP部分主要实现位置更新、呼叫、补充业务等业务中与VLR有关的处理;VLRDB部分主要负责移动用户基本数据及相关业务信息的存储和管理,并向业务的其它子系统提供访问数据的接口,是VLR系统中信息的载体和管理者,它不仅要有较高的执行效率、良好的稳定性与安全性,而且对实时性也提出了很高的要求。

内存数据库事务执行过程中要求没有内外存数据I/O操作,活动事务只与数据库的内存拷贝打交道,从而可以更好地满足VLR系统中业务处理对实时性的要求,因此,VLRDB子系统通常采用内存数据库作为其底层支撑[1]。内存数据库定义如下[2]:

定义1:对一个数据库DB,TS为数据库上所有事务的集合。$\forall T \in TS$, D(T)是事务T所操作的数据集($D(T) \subseteq DB$), DBM(t)为t时刻DB在内存中的数据集($DBM(t) \subseteq DB$), AT(t)是t时刻正在活动的事务集($AT(t) \subseteq TS$),若$\forall t$, $\forall T \in AT(t)$, $D(T) \subseteq DBM(t)$均成立,则称该数据库为内存数据库(Main Memory Database, MMDB)。

1 内存数据库特点

MMDB与硬盘数据库(Disk Resides Database, DRDB)的主要区别在于:MMDB的"主拷贝"常驻内存,而DRDB的数据访问涉及硬盘的I/O操作,内存读写速度比硬盘操作高出几个数量级,这使得MMDB能实现数据的实时、快速访问。MMDB中的数据是由CPU直接通过指针进行读写的,使得数据出错的可能性要高于DRDB,因此MMDB设计时要充分考虑数据读写异常的保护处理。

MMDB的数据库管理系统(Database Management

收稿日期:2007-03-25
作者简介:石英伟(1981-),男,河南洛阳人,硕士研究生,研究方向为现代数据库应用;姜 浩,博士,副教授,研究方向为工作流及数据库的应用。

System,DBMS)同样是内存数据库系统底层的核心,因而也具有普通数据库管理系统的一般功能:

* 永久数据管理,包括数据库的定义、存储、维护等,并且数据也应该保存到可靠的外存储介质中进行必要的备份。

* 为用户提供有效的数据存取、各种数据操作、查询处理、存取方法、完整性检查等接口。

* 事务的管理能力。包括并发时间调度、并发控制以及事务的执行管理。

* 为数据提供可靠的存取安全性检验。

* 可靠性恢复机制。在系统出现故障时,能保证迅速将系统恢复到正常工作状态。

MMDB 的管理系统与普通的数据库管理系统在数据库系统架构上也有着很大的差别:

(1)普通的数据库系统基本采用主存实例、数据库文件的方式,将数据库中的数据以文件的形式全部存放于磁盘等非主存系统设备上,而内存中的数据库实例只是以一定的调度算法调入实例的部分数据到系统内存中。

(2)内存数据库系统不仅要求磁盘系统上有数据库的文件存储备份,并且当数据库系统启动后,要求事务处理相关的全部数据都在系统内存中,同时要求系统内存中的数据与磁盘上的数据库文件采用一定的机制进行同步。

2　内存数据库核心设计

MMDB 的主要特点是"工作版本"常驻内存,即活动事务只与 MMDB 的内存拷贝打交道。因此,MMDB 的设计应该打破传统磁盘数据库的设计观念,充分利用内存直接快速存取的特点,重新设计内存数据的组织结构以实现 CPU 和内存空间的高效、安全利用[3,4]。

关系数据库具有结构单一、数据操作语言简单、表达能力强、用户使用方便直观等特点,所以利用关系模式设计内存数据库可以大大简化数据设计的复杂性,而且数据库也具有良好的可维护性。参考关系数据库基本原理,内存数据库核心主要负责管理简单的关系型数据,实现对数据的定义、描述、操作、维护等功能,接受并完成用户程序及数据库的不同请求,并负责保护数据免受各种干扰和破坏,提供数据访问的完整性、一致性、安全性机制,从而实现应用对数据的存取。

2.1　基于对象的内存关系模式

抽象性、封装性、继承性和多态性是面向对象方法的主要优点,采用面向对象的数据组织方式设计数据库,将存储信息的表实例有效地封装起来,访问者只有通过对象提供的基本方法才能接触到数据存储实体,从而有效保证了数据存储实体的安全性。基于对象的内存数据库管理系统的核心对象包括表、索引和队列,以及用于扩展功能的同步对象。基于对象的 MMDB 关系模式如图 1 所示。

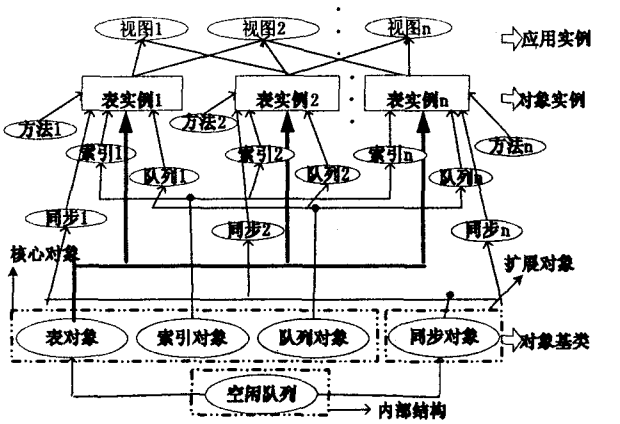

图 1　基于对象的 MMDB 关系模式

从图中可以看出,面向对象的内存数据管理分为三个层次[5]:

* 对象基类:对象基类中的核心对象是数据管理的基础,以核心对象为基础完成实例化,从而实现对数据的定义。

* 对象实例:该层利用第一层次提供的对象来完成实例化,赋予数据具体的含义。

* 应用实例:应用实例是面向具体应用的。以应用为宗旨,从对象实例中获取需要的数据,组成一个面向应用的数据视图,实现中对应于数据库的访问接口。

对于每一种基本对象类,都有相应的对象特征描述信息来规定对象的共有特性以及对象的基本方法,系统对对象的操作可以概括为三个部分:对象的注册定义、对象的方法定义、对象的方法重定义。

2.2　核心对象描述

表、索引和队列是 MMDB 的核心对象,都具有唯一的对象标识(Object Identifier,OID),OID 表示为二元组<Object Type,Object Index>,Object Type 表示对象类别,Object Index 表示对象在其所属类别数组中的序号。表、索引、队列对象都包含多个元素,为元素分配唯一的元素标识(Element Identifier,EID),EID 表示为二元组<OID,Element Index>,OID 是元素所属对象的标识,Element Index 表示元素在元素数组中的序号。索引对象和队列对象的创建都是为方便对表对象数据的管理和维护,一个表对象可以对应若干索引对象和队列对象。

表是关系数据库的核心,而记录是表中最主要的数据成员。记录在内存中的形式为一片连续的内存区

域,按照组成记录的字段顺序存放数据。为每条记录增加两个字节作为记录头,第一个字节描述记录的有效性,包含正常、已删除和待删除三种;第二个字节描述记录的失步信息,包含已失步和未失步两种。对记录进行存取时会判断记录有效性,在对记录进行同步时会参照记录的失步信息。关系表中各属性字段的描述信息在表定义时指出,内存中一条记录的实际长度为所有字段的长度之和加记录头长度 2。关系记录的逻辑结构如图 2 所示。

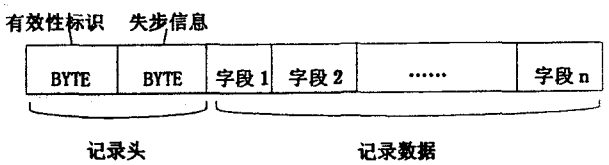

图 2 关系记录逻辑结构

采用面向对象的方法设计 MMDB 时,表对象应包含属性值和记录的操作方法,表对象的操作方法通过函数指针的引用来实现。表包含多个字段,所以表和字段存在聚合关系。表对象的操作方法有:

(1)基于表的操作:_CreateTable 创建表、Insert_R_TABLE 添加表记录等;

(2)基于记录的操作:_AppendTuple 申请空记录空间、_SetTuple 设置记录值、_GetTuple 查询记录值、_GetTupleAddr 获取记录首地址、_LocateTuple 定位记录、_DeleteTuple 删除记录等;

(3)基于字段的操作:_SetDomainByName 设置字段值、_GetDomainInfo 查询字段信息等。

索引的建立是为快速定位表中的记录,通常在关系表的主键上必须建立索引。索引中的基本数据是索引项,索引项描述索引关键字和对应记录位置的关系。MMDB 对索引提出的要求是占用内存小和维护代价低。一般的索引类型包括[6]:顺序索引、一维 HASH 索引、二维 HASH 索引以及散列 HASH 索引,应用时应根据索引字段变化的特点,选择合适的索引类型。索引对象操作方法分为索引的基本操作和哈希算法两类。

实现队列功能的目的是及时、有效地管理数据库的内存资源,队列是建立在表上的,按照队列的生成类型分为普通队列和有序队列,普通队列就是 FIFO 队列;有序队列中元素按照关键字的大小顺序排列。为加快队列的访问速度,设立队列为双向队列,每个队列项元素均有指向前后队列项的指针。队列对象同队列项对象存在聚合关系。

2.3 核心对象操作的实现

数据库启动后,首先需要进行一系列初始化工作,具体包括:读取系统配置、创建各个关系表、装载已有的表记录以及建立维护对应关系表所需的索引和队列。

数据库运行过程中,内存中的数据信息或索引信息会发生变化,现以表对象的基本操作为例,简要说明实现的方法流程,分四种情形:

(1)添加表记录。

系统操作生成一条新记录后,检查新记录的必选参数是否输入完整,调用函数 Insert_R_Table(lp_r_Tuple)进入添加记录的流程,lp_r_Tuple 标记新纪录的首地址;首先从新记录的必选参数中组合出表的主键(索引),通过函数_LocateTuple(hIDX_R_Table,(LPSTR)&TupleKey) 定位记录的主键是否有冲突,其中 hIDX_R_Table 为表对应的索引句柄,TupleKey 为表记录的主键结构体;在新记录主键没有冲突的情况下,通过函数_AppendTuple(hDB_R_Table)从句柄为 hDB_R_Table 的表的单向空闲资源队列中申请一条空白记录的内存空间,并标记该空间的对应的空记录为表的当前记录,之后通过函数_SetTuple (hDB_R_Table, (LPSTR)lp_r_Tuple)将表的当前记录改写为新记录 lp_r_Tuple,从而完成新记录在表结构中的添加;通过函数_InsertIndex(hIDX_R_Table, (LPSTR)&TupleKey)实现将新记录的主键信息加入表的管理索引中,以备实现记录的快速准确定位。

(2)修改表记录。

由于系统设计修改必须明确到表的某一条记录,因此系统要求输入的参数必须包含必选参数(主键),然后利用函数_LocateTuple 定位到相应的记录,利用函数_GetTupleAddr 获取记录首地址,并依据输入的其它参数修改记录对应的字段值,时间为 $O(1)$。

(3) 删除表记录。

通过系统输入要删除的信息,若可以组合到表的主键,则先利用索引方法快速定位到记录,然后删除记录,回收记录占用的空间到表对应的单向空闲资源队列中,以备以后新加记录使用,同时清除该记录对应的索引,时间为 $O(1)$;若依据输入的参数无法组合到主键,则需要使用遍历的方法,检查到符合删除条件的记录再删除,时间为 $O(n)$。

(4) 查询表记录。

系统输入要查询的信息,若可以组合到表的主键,则先利用函数_LocateTuple 的索引方法直接快速定位到记录,利用函数_GetTupleAddr 获取所要查询记录的首地址,转换到数据库接口出参中,返回系统所需的查询信息,时间为 $O(1)$;若依据输入的参数无法组合到主键,则需要使用遍历的方法,查询到符合条件的记录

再进行同样的处理,时间为 $O(n)$。

3 在 VLR 系统中的应用

VLR 主要管理其控制区内用户签约信息和位置信息,VLR 的实现基本上是 VLRDB 实时内存数据库的实现。VLRDB 主要用于存放用户相关的动态数据,同时为业务提供相关的访问和操作数据库数据的接口。

通过复合封装 MMDB 底层 DBMS 核心对象的各种数据操作方法,VLRDB 子系统向业务的其它子系统提供四类接口,分别描述如下[5]:

* DM 类接口:支持跨模块调用,使用该类接口的进程可以与数据库进程不在同一模块,依据接口入参 iParam 可划分为过程调用、同步调用、异步调用、异步无响应四种处理类型,满足不同模块间不同要求的数据访问,使用原语为:DBBOOL dbAccess(WORD EventNo, LPSTR iParam, LPSTR oParam),其中:EventNo 为操作请求事件号,iParam 为接口入参结构首地址,oParam 为接口出参结构首地址。

* CM 类接口:要求使用该类接口的进程必须与数据库进程在同一模块,不支持跨模块调用,使用原语为:VOID dbCall(WORD EventNo, LPSTR lpReq, LPSTR lpAck),其中:EventNo 为操作请求事件号,lpReq 为接口入参结构首地址,lpAck 为接口出参结构首地址,dbCall 接口是一个纯粹的过程调用,依据事件号,先判断事件类别,进入相应的过程集,调用过程集中的一个过程函数,在过程函数中依据请求类别来操作内存数据。

* EV 类接口:通过数据库向其它进程发送相应消息来实现。

* 全局量接口:使用简单,原则上不允许调用者修改全局量的值。

VLR 系统设计中,VLRDB 与 VLRMAP 在同一模块,因此它们之间的接口设计为内部接口,主要通过 CM 类接口的函数调用来实现,少部分通过内部消息传递来实现。而 VLRDB 同 MSC 的其它部分(比如:BSSAP+、MM、MCC、SS、TM)属不同模块,因此向这些模块提供的接口主要通过 VLRMAP 实现,主要通过 DM 类接口的函数调用来实现,因此同 VLR 系统交互的模块实际上是同 VLRMAP 模块协商来实现对业务所需内存数据的实时访问。

4 性能测试

为验证 MMDB 的实时性比 DRDB 有质的提升,下面给出所作试验的初步结果。图 3 为连续操作记录数与消耗时间的关系,试验结果表明 MMDB 同 DRDB 有相当显著的时间消耗差别。随着连续操作记录数的不断增加,DRDB 的时间消耗增加幅度非常明显,而 MMDB 的时间消耗增加幅度要小得多,基本呈线性关系,从而说明 MMDB 的实时性能有质的突破。

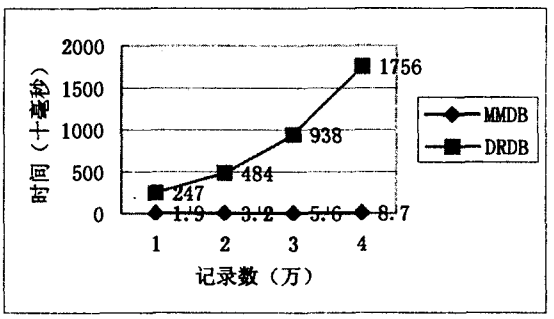

图 3　MMDB 与 DRDB 的性能比较

5 结束语

应用于 VLR 系统的 VLRDB 数据库子系统采用面向对象的设计方法,有效实现移动交换机中实时数据的存储和维护,是交换机系统中业务相关动态信息的存储载体和数据操作、维护的良好工具,实际应用中也同样表现出较强的灵活性、适应性、稳定性和实时性。基于此,实用内存数据库系统在通信领域的应用尤为突出,尤其是在访问位置寄存器(VLR)、归属位置寄存器(HLR)、移动通信网络侧的短消息中心等移动实体中,为高层的业务处理和系统扩容提供了快速的数据服务,因而也推动了应用领域的飞速发展。

参考文献:

[1] DeWitt D J, Katz R H, Ohlen F, et al. Implementation techniques for main memory databases[C]//In proceedings of ACMSIGMOD 1984 International Conference on Management of Data. Washington, D.C.:[s.n.],1984:1-8.

[2] 刘云生. 现代数据库系统[M]. 北京:国防工业出版社,2001.

[3] 王洪海,潘朝华. 内存数据库的数据结构分析[J]. 现代电子技术,2004(3):99-101.

[4] 刘云生,潘　琳. 实时数据库系统的内存数据库组织与故障恢复[J]. 小型微型计算机系统, 2001, 20(5):37~43.

[5] 3G 平台数据库项目组. 3G 平台数据库核心子系统设计[R]. 深圳:中兴通讯,2006.

[6] Lehman T, Carey M. A study of index structure for MM-DBMS[C]//Proc of the 12th International conference on Very Large Database, VLDB. Kyoto, Japan:[s.n.],1986:294-303.