文章编号:1000-5641(2014)05-0311-09

内存数据管理技术在族谱信息系统中的应用

张文杰, 彭智勇, 彭煜玮
(武汉大学 计算机学院, 武汉 430072)

摘要: 设计并实现了具有数据录入、数据服务、数据输出功能的族谱信息系统. 族谱信息系统采用了分布式结构,在每个分布数据节点引入内存数据管理技术,采用列存储模型,自动初始化热点数据,并根据用户请求组织数据建立索引,同时利用事务日志对每个分布数据节点的内外存进行数据同步,对中心数据节点和分布数据节点进行数据同步.

关键词: 族谱; 分布式结构; 内存数据管理技术; 索引; 数据同步
中图分类号: TP392 **文献标识码**: A **DOI**:10.3969/j.issn.1000-5641.2014.05.028

Application of in-memory data management technology in genealogy information system

ZHANG Wen-jie, PENG Zhi-yong, PENG Yu-wei
(*School of Computer, Wuhan University, Wuhan* 430072, *China*)

Abstract: In this paper, a new genealogy information system was designed and implemented. It provides data inputting, data service and data outputting functions. The new genealogy information system is based on the distributed structure. Its distributed nodes employ the in-memory data management technology. Every distributed node initializes the hot data and creates index based on the user request in main-memory column-stores. And it implements the data synchronization between the disk and in-memory as well as the data synchronization between distributed nodes and data center data node with transaction logs.

Key words: genealogy data; distributed structure; in-memory data management technology; indexing; data synchronization

0 引 言

族谱又称为家谱、宗谱,是一种记录家族世代繁衍和重要人物事迹的图文体裁. 族谱文献对于我们了解人文历史有很大的帮助,并且在政治经济学、地理学、群体遗传学等方面都有着潜在的研究价值[1]. 传统的中国式族谱通常以纸质、布质等形式的谱书为承载形式,各

收稿日期:2014-06
基金项目:国家 863 课题(2012AA011004);国家自然科学基金重点项目(61232002)
第一作者:张文杰,男,在读硕士生,研究方向内存数据管理. E-mail: zwjwhu@whu.edu.cn.
第二作者:彭智勇,男,教授,博士生导师,研究方向数据管理. E-mail: peng@whu.edu.cn.
通信作者:彭煜玮,男,博士,研究方向数据管理. E-mail: ywpeng@whu.edu.cn.

族、各家拥有自己独立的谱书. 这些谱书在存在形式上相互独立,但是在内容上却有很强的相关性. 传统的谱书式族谱在信息交互和共享方面存在着先天不足,且实体的谱书不便于后续的修改(续修)以及保存. 因此,近年来族谱数字化受到了社会的广泛关注.

数字化族谱系统除了能提供给用户录入族谱信息、利用族谱信息的功能之外,还需要兼顾中国式族谱的特殊需求——谱书. 家族在编修族谱时,通常都需要以采集好的族谱数据为基础,通过编辑、排版形成内容丰富、图文并茂、样式美观的谱书,然后将其印刷成册并分发给族人. 目前,国内外在数字化族谱系统方面已有一些研究和开发工作. 著名的族谱网站 FamilySearch[2] 中,用户可以方便地创建和管理个性化族谱空间,但没有提供族谱数据纸质化输出功能. 文献[3-4]都实现了基于单机形式的族谱录入软件:将族谱的制作工作分割成多个任务,由多位制作人员分别完成这些任务并以文件存储任务中的族谱数据,最后将多个数据文件合并,编辑形成最终的族谱. 但是这种数据管理方式不利于信息的共享,而且多个数据文件之前存在较多的冗余以及冲突,无法自动完成数据文件合并. 为了加快族谱数字化进程,结合实际的应用需求,本文设计并实现了一个基于 B/S 架构的族谱信息系统,该系统提供了族谱数据的录入、查询服务、纸质化输出等功能. 结合族谱信息系统中的实际需求,本系统的数据存储采用了分布式结构和内存数据管理技术,大大加快了族谱信息录入和检索的速度.

本文内容组织如下:第 1 节介绍族谱信息数据的特点;第 2 节介绍系统功能,并分析引入内存数据管理技术的必要性;第 3 节将简单介绍本系统中所涉及的相关内存数据管理技术;第 4 节介绍内存数据管理技术在族谱信息系统中的具体应用;第 5 节对本文工作进行总结并对未来工作进行展望.

1 族谱数据简介

族谱数据通常由三部分构成:世系数据、文档数据以及多媒体数据.

(1) 世系数据包括人物数据和人物之间的亲缘关系数据. 其中亲缘关系主要分为父子(父女)、母子(母女)、配偶、过继(从亲属中收养子女)、兼祧(一位男子同时继承多家). 世系数据是族谱数据中最重要的部分.

(2) 文档数据包括在族谱中家族大事件的文字记录和重要的人物传记,族谱中的文档数据除文本之外还包含图表等,呈现一种图文并茂的形式.

(3) 多媒体数据包括族谱中记录人物或者家族大事件的图像、音频、视频,这里的多媒体数据并不包括文档数据中含有的图表.

族谱数据中的文档数据、多媒体数据和一般的文档数据、多媒体数据在组织和存储形式上并无明显区别,而世系数据作为族谱数据中的核心数据,其特点鲜明.

如果把世系数据中每个人物作为节点,人物之间的关系作为边,则世系数据就构成了一种类似树的结构,下文中也把这种结构称为世系树,如图 1 所示.

如果考虑更广泛的亲缘关系(如联姻),很多世系数据就会联系在一起构成类似森林的结构,这和社会网络[5]数据非常相似. 但是世系数据比社会网络数据所表达的人物群体更加特定,人物之间的关系更加明确.

总结起来,世系数据具有以下特点:

(1) 表达对象是特定的人物"群体"——家族(或者支系),人物关系是亲缘关系,相比其

他人际关系要更加紧密和牢固.

(2) 如果把人物视作节点,把人物之间的关系视作边,世系数据构成了一种特殊的层次结构—世系树.

(3) 如果把多个世系树用联姻关系联系在一起,会构成一种特殊的图结构,类似于森林和社会化网络数据.

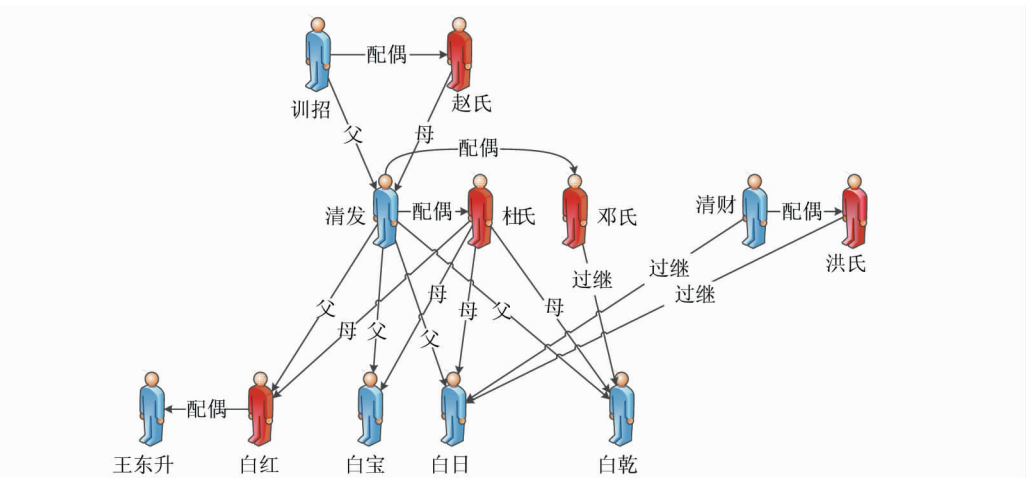

图 1　世系树示例图

Fig. 1　The tree structure of genealogy data

2　族谱信息系统

传统族谱信息系统采用的是单机管理不易于扩展功能,数据分散且有大量冗余,无法利用这些族谱数据向公众用户提供服务. 为了更好地收集和利用族谱数据,我们设计并开发了一款基于 B/S 架构的族谱信息系统. 该系统支持多用户并行录入同一族谱中的数据,并统一对族谱数据进行管理,同时通过本系统还可向公众用户提供对已录入族谱数据的检索.

族谱信息系统的主要功能包括数据录入、数据服务和数据输出.

2.1　数据录入

数据录入功能主要包括三部分:世系数据录入、文档数据录入、多媒体数据录入. 其中世系数据录入是指录入人物的基本数据以及录入人物之间的联系数据.

2.2　数据服务

数据服务功能主要包括数据展示、统计检索、一键寻祖和一键寻亲.

(1) 数据展示

数据展示功能包括族谱展示和对照预览. 族谱展示功能主要显示一个族谱的基本信息. 对照预览则用于在正式输出纸质化族谱之前以各种不同的样式来预览输出的效果.

(2) 统计检索

统计检索提供了对系统中的族谱数据进行统计和检索的功能. 族谱统计是显示整个族谱的统计信息,如总人数、男女比例、生死状况等. 简单检索和组合检索是在某些族谱属性或者人物属性上进行检索的功能.

(3) 一键寻祖和一键寻亲

一键寻祖可以从族谱中根据人物之间的亲缘关系得到指定人物在指定范围内(比如限定世代数)的祖先. 一键寻亲则是指在族谱数据中找到两个指定人物之间的亲缘关系链(即两人是通过哪些人关联在一起).

2.3 数据输出

为了满足用户传统纸质谱书的需求,族谱信息系统中提供了数据输出功能,主要包括族谱编排、族谱生成.

(1)族谱编排

谱志编排功能是通过用户的个性化需求对谱书的样式、数据出现的顺序、名词的表达方式等进行设置.

(2) 族谱生成

族谱生成功能主要是在族谱编排过后,按照用户的个性化需求从原始族谱数据中转换生成电子版的谱书以供印刷.

2.4 应用内存数据管理技术的必要性

通过对族谱信息系统的功能分析,族谱信息系统具有以下特点.

(1)数据源单一. 在数据录入过程中,对每个录入用户来说,只能操作自己参与录入的族谱数据. 同时,数据输出过程中,只需要去访问要生成电子族谱的特定族谱的数据.

(2)数据量较大. 每个族谱都存有数量和容量庞大的世系数据、文档数据、多媒体数据.

(3)实时性要求高. 由于本系统基于 B/S 模式设计,无论是数据录入、数据服务还是数据输出功能,系统响应时间都应该很短.

族谱信息系统的大数据量和高实时性的特点对系统实现提出了挑战. 而随着主存的成本显著降低,许多成熟的内存数据管理技术为族谱信息系统的实现提供了解决方案[6].

3 内存数据管理技术

内存数据管理[7]的关键技术包括存储结构[8]、索引结构[9]、并发控制[10]、同步策略[11]、故障恢复[12]等. 本节将重点介绍系统中用到的索引结构和同步策略.

3.1 索引结构

内存数据库由于其工作的主版本保存在内存中,所以内存数据库的索引选择应结合存储介质的特点,从而通过索引的建立来保证内存数据库查询操作的高效性. 目前在内存数据库中经常选用的索引结构有 hash 索引和 T 树索引.

(1) hash 索引[13]定义了一个 hash 函数,通过将关系表的索引项传入到 hash 函数可以计算出相应的 hash 值,从而在索引项和 hash 值之间建立起对应关系,通过 hash 索引查找数据只需常数时间的复杂度.

(2) 在内存数据库中目前较广泛使用的一种树是结合 B 树[14]和 AVL 树进化而来的 T 树[14]. T 树的单个节点有多个数据,因此拥有良好的修改和存储特性. 由于 T 树属于 AVL 树的一种演进,具有 AVL 树的平衡特性,从而进一步提升了树的搜索性能. 因此 T 树在时间和空间两者间具备较好的平衡性.

hash 索引在进行定值的查找时效率很高,而 T 树索引一方面具有树的二叉性而且其设计符合内存数据库存储介质的特性,所以当前主流的内存数据库都至少提供了这两种索引结构.

3.2　同步策略

内存数据管理的数据同步更新技术大致可以分为表复制技术、事务复制技术、触发器技术和影子表技术. 这里主要介绍表复制技术和事务复制技术[15].

(1) 表复制技术:采用把某一时刻源数据表的内容通过网络发送到复制的副本,因为复制的内容是表的某一时刻的状态,所以又被形象地称为表快照. 表快照的复制不是以事务为基础,所以副本缺乏基本的关系完整性. 基于表复制技术不需依赖特别的机制,不占用额外的系统资源,管理和操作也非常容易,而且在同步初始化和崩溃恢复时是必须的. 但是全表更新效率很低.

(2) 事务复制技术:事务复制技术是把修改源数据的事务通过网络发送到复制的副本,复制可以是修改的表项事务或事务日志. 复制的时间可根据应用需求而确定. 副本接收到复制内容后,要重复一遍接收到的事务操作来实现与数据源的一致. 一般是基于数据库日志通过分析日志的信息来获得数据的差异,最后达到数据同步.

4　内存数据管理技术在系统中的应用

在族谱信息系统中需要大量的递归查询操作,而且系统对数据存取的实时性要求比较高,只依靠基于磁盘的传统数据库系统无法满足族谱信息系统的要求. 为了保证数据处理的实时性和可靠性,族谱信息系统中采用内存和外部存储设备(如磁盘)共同作为数据的存储介质. 族谱信息系统将实时或关键性数据的操作放在内存数据库中进行,由于内存的数据存取速度比磁盘快,引入内存数据管理技术会使族谱信息系统更高效,更迅捷.

4.1　族谱信息系统结构

在族谱信息系统的业务逻辑中,系统的运营商负责分配录入任务给各个代理商,各代理商再将任务分割为多个子任务,并组织多位录入人员进行录入. 为了增加子系统的可靠性和灵活性,族谱信息系统采用分布式结构[16](如图 2 所示). 系统中包括了一个中心数据节点和多个分布数据节点. 中心数据节点储存了所有的族谱数据;而每个代理商拥有一个分布数据节点,存储了本代理商代理录入完成的族谱数据.

每个分布数据节点都是由一个磁盘数据库和一个内存管理单元组成. 其中内存管理单元包括内存数据库、用户请求处理模块、接收队列、发送队列、节点状态管理模块、资源管理模块和数据同步模块,内存数据库采用列存储模型来实现存储. 用户请求处理模块主要是接收用户请求,并根据用户请求进入不同的处理分支. 接收队列用于接收用户提交的新增的数据或者是修改的数据. 发送队列用于发送给用户所要求的查询结果数据. 数据同步模块用来维持分布数据节点和中心数据节点的数据同步. 资源管理模块主要是对内存资源进行分配和回收.

4.2　数据录入功能的内存管理策略

在数据录入功能中,每个录入用户登录之后首先会选择自己要录入的族谱,系统自动连接该族谱隶属的代理商的分布数据节点.

分布数据节点中的用户请求处理模块会识别用户数据录入请求,把用户选择的族谱数据作为热点数据存入到内存数据库当中. 由于数据录入功能的主要操作是插入新元组,所以元组集合在内存数据库中采用堆组织以便高效地插入新元组;同时按照键值建立哈希索引,其中重名的会依次放在指针数组中.

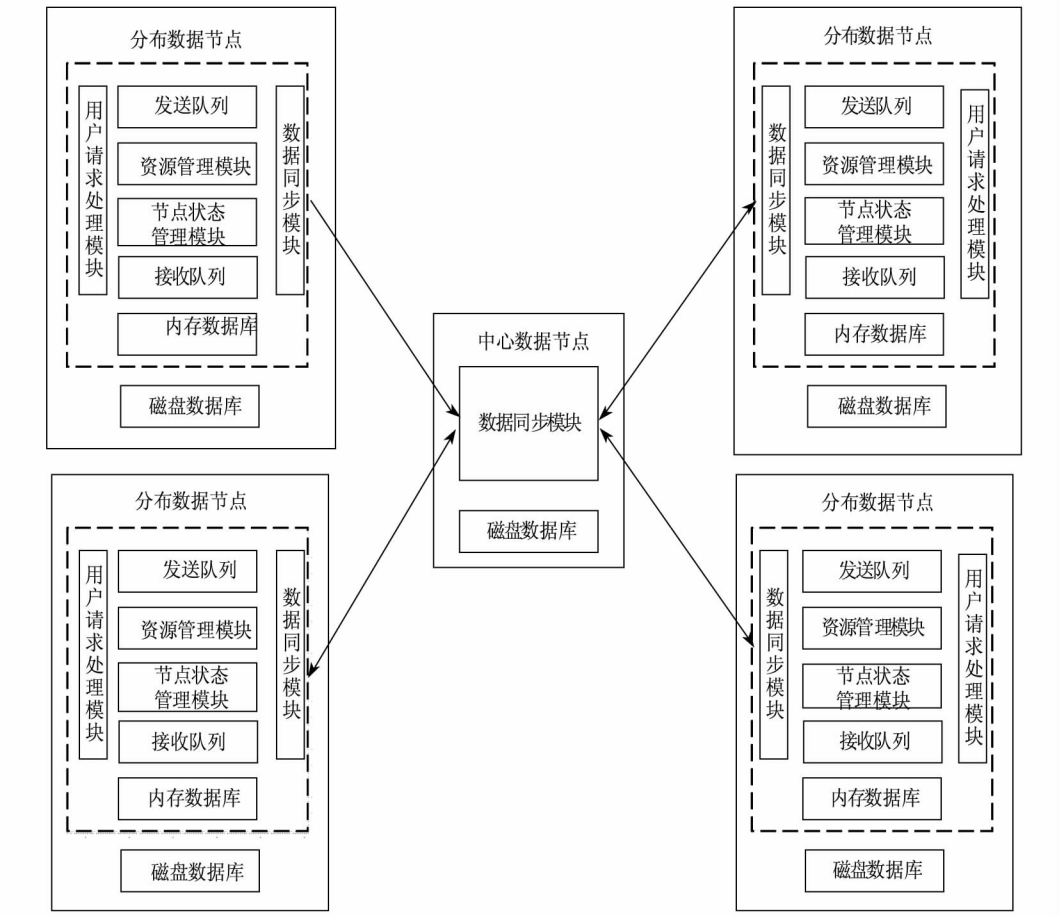

图 2 族谱信息系统架构

Fig. 2 The structure of the genealogy information system

当用户插入新元组时,新元组会加入到系统的接收队列,分布数据节点会把新增元组加入到内存中存储; 当用户需要请求检索数据时,对应分布数据节点会根据索引快速定位数据位置并返回给用户;当用户修改数据时会把修改后的数据加入到系统的接收队列,分布数据节点会把接收队列中的数据依次更新;当用户删除数据的时候,分布数据节点会通过索引定位到该数据并执行删除,同时内存资源管理模块进行内存资源回收.

4.3 数据服务功能的内存管理策略

由于在数据服务功能中主要是对世系数据的大量结构化查询,分布数据节点采用 T 树索引结构来存储元组信息. 其中每个节点的数据中都含有人物对象的详细信息和分别指向父亲、母亲、过继或兼祧父亲、过继或兼祧母亲的四个指针. 同时建立哈希索引(同第 4.2 节).

当数据录入导致本族谱世系数据发生改变的时候,会对内存 T 树索引和哈希索引进行更新,分为以下几种情况.

(1) 如果需要增加新的元组,系统会对 T 树索引做插入操作并更新哈希索引;

(2) 如果需要更新某个元组,系统会通过 T 树索引找到旧元组直接进行更新如需要则

同时更新哈希索引;

(3) 如果需要删除某个元组,系统会先通过 T 树索引找到该元组然后删除该元组,更新 T 树索引和哈希索引,同时资源管理模块进行内存资源回收.

在数据服务功能中,当用户请求一键寻祖的时候输入要寻祖的人物谱名和祖先的世代数(可以不输入,默认为族谱中最小的世代数),分布数据节点会首先通过哈希索引找到 T 树索引中对应的人物对象元组节点. 如果存在多个重名的人物对象则会返回几个人物对象的具体信息供用户选择;如果该谱名只对应一个人物对象或者用户从重名人物中选择了一个人物对象,则分布数据节点会从世系树中该人物对象节点开始循环地通过父亲或兼祧父亲指针寻找祖先节点;当该祖先的世代数等于用户输入的值,则停止循环并返回该祖先节点元组给用户.

4.4　数据输出功能的内存管理策略

在数据输出功能中,每个用户需要首先选择族谱,分布数据节点把对应的族谱数据作为热点数据载入到内存数据库中,在后续的族谱编排和族谱生成中可以直接访问分布数据节点的内存,并建立 T 树索引(同第 4.2 节).

当用户对世系数据进行分组(可以按照个人或者是世代分组)时,分布数据节点会通过 T 树索引检索对应人物更新其分组号;当用户按照需求对世系和文档进行分卷时,会对世系分组和文档数据进行排序用于生成对应的电子族谱,同时分布数据节点会按照用户的编排顺序对内存中的数据进行排序;当用户选择相应的模板请求族谱生成的时候,分布数据节点会根据模板的格式生成族谱并返回给用户.

4.5　数据同步策略

在族谱信息系统中,每个分布数据节点的内存数据库保存着实时数据,但是内存属易失性存储,为了提高数据的可靠性,必须和外存数据库进行数据同步. 同时中心数据节点作为所有分布数据节点族谱数据的副本也需要和分布数据节点进行数据同步.

4.5.1　分布数据节点内外存数据同步

分布数据节点内外存数据同步属于单向同步,除了内存数据初始化外,数据都是从内存数据库传输到外存数据库当中.

在族谱信息系统中,分布数据节点的内外存数据同步是由常驻后台进程 MMSyn 来实现的. 分布数据节点启动后,MMSyn 进程就会自动启动. MMSyn 进程启动时需要初始化系统设置的同步周期时间和进程数阀值. MMSyn 进程在上次同步操作完成和下次同步操作开始之间会休眠一个同步周期. 每次 MMSyn 进程被唤醒之后,会通过事务日志来检测是否存在数据更新,如果有而且当前的进程数低于阀值就进行数据更新,否则 MMSyn 进程继续休眠. MMSyn 进程在数据更新过程前会读取存储的上次完成同步的事务日志序列号,从下一事务日志开始在外存里重做事务操作从而完成同步.

4.5.2　分布数据节点与中心数据节点的数据同步

分布数据节点分散存储着各个代理商代理录入的族谱信息,而中心数据节点作为稳定的中心数据备份必须和分布数据节点进行数据同步. 分布数据节点与中心数据节点的数据同步属于单向同步,除了分布数据节点崩溃从中心数据节点恢复以外,数据都是从分布数据节点传输到中心数据节点当中.

在族谱信息系统中,分布数据节点与中心数据节点的数据同步是由分布数据节点常驻

后台进程 DSyn 和中心数据节点常驻后台进程 CSyn 来实现的. 在族谱信息系统启动后,CSyn 和 DSyn 进程会自动启动.

和 MMsyn 进程相似,DSyn 进程在启动时需要初始化系统设置的同步周期时间和进程数阀值. 每次 DSyn 进程被唤醒之后,会读取存储的上次完成同步的事务日志序列号 n,如果当前最大的日志序列号 m>n(日志序列号是递增的),则将 n<日志序列号>m 的日志发送给中心数据节点.

中心数据节点一旦启动,就会开启 CSyn 进程. 当分布数据节点发送过来日志序列时,CSyn 重做事务操作从而完成同步.

5 结 论

本文设计并实现的族谱信息系统采用了 B/S 架构,能更好地支持族谱数据的分散录入以及集中共享的现实需求. 在族谱数据管理方面采用了分布式结构,其中包括中心数据节点和分布数据节点. 中心数据节点存储全部族谱的数据,分布数据节点存储对应代理商录入的族谱数据,通过同步策略实现中心数据节点和各个分布数据节点的数据同步,大大加强了系统的可靠性和灵活性.

分布数据节点引入了内存数据管理技术,采用列存储模型存储结构,并根据用户具体的请求初始化热点数据,建立索引. 用户的操作在分布数据节点内存中进行,加快了系统的响应速度. 同时,系统利用事务日志进行分布数据节点的内外存同步和内存数据库恢复,增强了系统的可靠性.

未来的工作还需要考虑热点数据的优化选择、分布数据节点负载均衡等问题.

[参 考 文 献]

[1] 张卓. 开发利用族谱档案的意义[J]. 云南档案,2006(3):32-33.

[2] FamilySearch[EB/OL]. http://familysearch.org.

[3] 启航宗谱[EB/OL]. http://www.qhzprj.com.

[4] 中根网[EB/OL]. http://www.zongen.com.

[5] GRANOVETTER M. Economic action and social structure: the problem of embeddedness[J]. American Journal of Sociology, 1985,19(3): 481-510.

[6] FREITAS R F, WILCKE W W. Storage-class memory: The next storage system technology[J]. IBM Journal of Research and Development, 2008, 52(4/5): 439-447.

[7] LEHMAN T J, CAREY M J. A study of index structures for main memory database management systems[C]// Conference on Very Large Data Bases. 1986, 294.

[8] ABADI D J, MADDEN S R, HACHEM N. Column-stores vs. row-stores: how different are they really? [C]// Proceedings of the 2008 ACM SIGMOD international conference on Management of data. ACM, 2008: 967-980.

[9] GRAEFE G, IDREOS S, KUNO H, et al. Benchmarking adaptive indexing[M]//Performance Evaluation, Measurement and Characterization of Complex Systems. Berlin: Springer, 2011: 169-184.

[10] ALCANTARA D A, SHARF A, ABBASINEJAD F, et al. Real-time parallel hashing on the GPU[C]//ACM Transactions on Graphics (TOG). ACM, 2009, 28(5): 154.

[11] DEWITT D J, KATZ R H, OLKEN F, et al. Implementation techniques for main memory database systems[M]. ACM, 1984.

[12] OUSTERHOUT J, AGRAWAL P, ERICKSON D, et al. The case for RAMClouds: scalable high-performance storage entirely in DRAM[J]. ACM SIGOPS Operating Systems Review, 2010, 43(4): 92-105.

[13] LEHMAN T J, CAREY M J. A study of index structures for main memory database management systems[C]// Conference on Very Large Data Bases. 1986, 294.

[14] LU H J, Yuet Yeung Ng, Tian Z P. T-tree or b-tree: Main memory database index structure revisited[C]//Database Conference, 2000. ADC 2000. Proceedings. 11th Australasian. IEEE, 2000: 65-73.

[15] LEE S W, MOON B. Design of flash-based DBMS: an in-page logging approach[C]//Proceedings of the 2007 ACM SIGMOD international conference on Management of data. ACM, 2007: 55-66.

[16] KALLMAN R, KIMURA H, NATKINS J, et al. H-store: a high-performance, distributed main memory transaction processing system[J]. Proceedings of the VLDB Endowment, 2008, 1(2): 1496-1499.

(责任编辑　李　艺)

(上接第 289 页)

[参　考　文　献]

[1] Stored Procedure[EB/OL]. http://en.wikipedia.org/wiki/Stored_procedure.

[2] PL/pgSQL[EB/OL]. http://www.postgresql.org/docs/8.3/static/plpgsql.html.

[3] OceanBase[EB/OL]. http://alibaba.github.io/oceanbase/.

[4] 杨传辉. 大规模分布式存储系统原理解析与架构实战[M]. 北京:工业出版社,2013.

[5] 彭智勇,彭煜玮. PostgreSQL 数据库内核分析[M]. 北京:机械工业出版社华章公司,2012.

[6] AHO A V, ULLMAN J D. Principles of Compiler Design[M]. [s.L.]:Addison-Wesley, 1977.

[7] STONEBRAKER M, KEMNITZ G. The Postgres Next Generation Database Management System[J]. Commun ACM, 1991, 34(10): 78-92.

[8] KALLMAN R, KIMURA H, NATKINS J, et al. Abadi: H-store: a high-performance, distributed main memory transaction processing system[J]. PVLDB, 2008,1(2): 1496-1499.

[9] STONEBRAKER M, WEISBERG A. The VoltDB Main Memory DBMS[J]. IEEE Data Eng Bull, 2013, 36(2): 21-27.

[10] STONNEBRAKER M, MADDEN S, ABADI D J, et al. The end of an Architectural Era: (It's Time for a Complete Rewrite)[C]//VLDB '07: Proceedings of the 33rd International Conference on Very Large Data Bases, 2007: 1150-1160.

(责任编辑　王善平)