一种 Shared-Nothing 分布式数据库的构建方法

龙 源,郑 彦
(南京邮电大学 计算机学院,江苏 南京 210003)

摘 要:分布式数据库能够以相对低廉的代价满足海量数据分析处理的性能需求,兼具良好的可扩展性。文中采用 Shared-Nothing 架构及 MySQL 数据库来实现分布式数据库访问层的功能架构及模块设计,提出了使用列存储机制提升分布式数据库系统查询性能的思路,包括数据存储的方法及策略。对所设计实现的系统进行了基准性能测试及扩展性能测试,结果表明,文中所实现系统相较传统列式数据库具有出色的查询性能表现,同时具有优秀的可扩展性,能够以较低的代价满足海量数据分析处理所带来的额外数据库性能提升要求。

关键词:分布式数据库;Shared-Nothing;MySQL;列存储
中图分类号:TP31 **文献标识码**:A **文章编号**:1673-629X(2012)10-0079-04

A Construction Method of Shared-Nothing Distributed Database

LONG Yuan, ZHENG Yan
(College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract: Distributed database can meet the performance demands of massive data analysis and processing with relatively low cost, with good scalability. It uses Shared-Nothing architecture as the overall system architecture and the MySQL database architecture to achieve module design of distributed database access layer functions. Put forward the idea of using the column storage mechanism to enhance the query performance of distributed database systems, including data storage methods and strategies to achieve them. It conducted the testing for the designed system from the benchmarks and scalability. The performance results show that this system is excellent in query performance and scalability. This system would meet the demand of the additional database performance upgrading requirements brought by massive data analysis with lower cost.

Key words: distributed database; Shared-Nothing; MySQL; column-stores

0 引 言

随着以 WEB 应用为主的网络时代的到来,大规模运算及高负载应用急剧增长,大量的应用及数据处理均由服务器端来支撑,这对数据库系统的性能挑战越来越严峻。传统的集中式数据库系统在面对海量的数据增长及应用事务请求时,很难保持高效的表现。在数据挖掘分析领域,大量的复杂组合数据查询也对传统的行式数据库性能提出了挑战。

行式数据库系统[1~3]能够很好地处理那些插入删除数据的事务,但是列式数据库能更好地处理那些查询表中某些列信息的事务请求。检索同样的数据,行式数据库系统平均消耗的物理 I/O 资源是列式数据库系统的 5~10 倍[4,5]。

收稿日期:2012-02-17;**修回日期**:2012-05-23
基金项目:国家"863"高技术发展计划项目(2006AA01Z201)
作者简介:龙 源(1989-),男,硕士研究生,研究方向为数据仓库与决策支持系统;郑 彦,教授,研究方向为数据仓库与决策支持系统。

分布式技术可以将各个节点分散的物理资源纳入到整体系统中,通过负载均衡、任务的分拆与运算结果的组合等方法[6],对大型任务进行分解并充分地利用每个节点的物理资源以达到整体资源最优化[7,8]。

分布式结构 Shared-Nothing 因其良好的可扩展性,在 Web 应用开发中广受欢迎。文中针对以上不足,基于 Shared-Nothing 分布式架构,依托 MySQL 数据库,实现其分布式列式数据库及其访问层架构,使之能够适应当前海量数据的分析查询性能需求。

1 高性能分布式计算架构概述

高性能分布式计算中架构被经常使用的有如下三种架构:

1) Shared Memory 架构是在多台相同型号的多处理器机器上实现的,多个 CPU 共享共同的物理内存及一个单独序列物理存储,多个 CPU 在一定的通信机制下通过共享的物理内存进行通信,将运算任务的性能负载均衡分配到各个计算核心上来提高运算性能。

2) Shared Disk 架构的每个独立节点拥有自己的

处理器和内存,这些节点都接入同样的物理存储序列。通常在存储局域网络系统(storage area network system)及网络存储系统(network-attached storage system)中使用。

3) Shared-Nothing 架构是一种分布式计算架构,其中的每一个节点都是独立、自给的,物理资源不进行共享,整个系统中都不存在单点竞争。最特别的是,所有节点都不互相共享内存及硬盘存储。Shared-Nothing 架构系统通常将它的数据分割并存储在众多节点上的不同数据库中(分配不同的终端来处理不同的事务),或者通过某些协议使每个节点保持它们自己的应用数据副本。基本结构如图 1 所示:

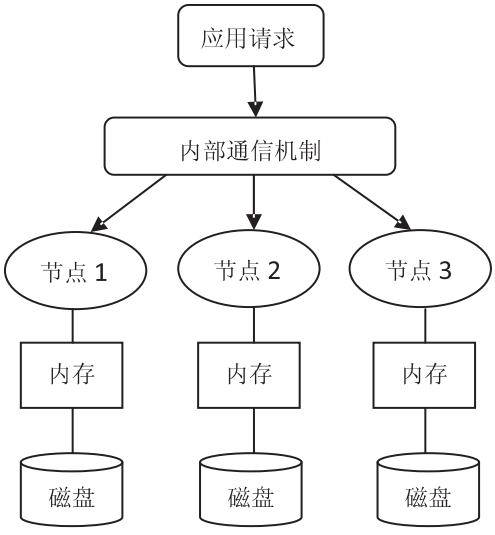

图 1 Shared-Nothing 架构简示

Shared Memory 架构由于其受到总线带宽的限制,二级缓存需要同步,硬件复杂性过高,难以布置在大规模数据库集群中,在分布式数据库系统中较少使用。

Shared-Nothing 架构还可以由廉价的普通 PC 和网络硬件来搭建,Google、Amazon、Yahoo 和 MSN 都证明了这一点。据报道,Google 搜索的支撑集群就是由上万台普通的 PC 充当 Shared-Nothing 架构的节点。总的来说,Shared-Nothing 架构相比 Shared Disk 架构以其出色的可扩展性占据了明显的优势。

2 列式数据库概述

列式数据库(Column-oriented Database)是采用列存储(Column Store)架构实现数据存储功能的数据库。每一列数据都会连续地存储在硬盘的特定区域,通常通过使用大磁盘分页(large disk pages)或大读取单元(large read units)来分担在硬件系统中扫描多列所带来的磁盘磁头寻道(disk head seek)开销[9]。为了改善读取性能[10],数据列中的数据通常都十分密集,同时显式存储记录 ID 并尽可能使用压缩比高的压缩策略[11]。列扫描器与行扫描器的区别在于列扫描器需要将数据位置信息转换为磁盘地址,在得到不同列数据后根据需要组合或重建出部分或整个元组的数据。通过接受重建好的元组,连接操作器也可以基于列扫描器,或者其可以直接处理连接索引来获取对应数据。列式数据库的优势体现在数据的压缩存储及查询方面,而由于存储数据的结构设计,对数据的操作上性能并不是十分出色[12]。对于海量数据下大量复杂查询检索的数据仓库应用情况来说,列式数据库是一个较为合适的存储解决方案。

3 Shared-Nothing 分布式数据库的架构及实现

3.1 整体架构设计

将基于 Shared-Nothing 分布式数据库架构设计为三层,如图 2 所示:

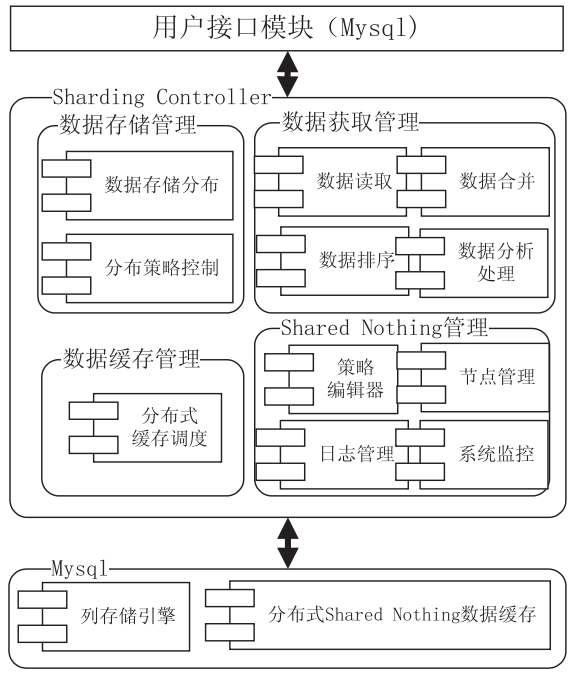

图 2 系统整体结构关系

第一层:用户接口模块。负责提供用户接入并访问数据库的接口,管理数据库的连接线程及线程池,将来自用户的操作及查询请求发送给分布式数据库控制器(Sharding Controller),并将从分布式数据库控制器获取到的操作响应及查询数据返回给用户。

第二层:分布式数据库控制器模块。它是负责控制管理整个分布式数据库运行的控制模块,其中包括数据存储管理、数据获取管理、数据缓存管理、数据分布式管理等子模块。它能够将来自用户接口模块的请求通过分布式策略将数据库操作或查询任务分配到各个节点上,并对返回的数据进行整合排序等处理以满足查询要求。

第三层:MySQL 存储层。它主要负责数据库中数据的物理存储,文中采用了列存储引擎及分布式数据缓存,试图优化数据库在海量数据查询分析应用环境下的表现。

采用该结构设计有如下优点:

a. 性能提升。采用分布式架构可以有效地提高数据库系统的性能,同时也提供了系统的可扩展性,使得系统对数据存储及查询分析要求的性能增长可以通过增加节点来满足。

b. 层间独立。通过三层模型来实现整个数据存储系统,层间使用标准接口通信,这使得每层可以选取最适当的策略而不用考虑是否会影响到其它层的功能实现。

c. 数据安全。在三层模型中,用户不会直接访问到数据存储层,也就无法对数据进行直接操作,可以在管理层设计数据安全策略来避免误操作及恶意操作等来自用户对数据的威胁,这可以大大提升数据的安全性。

d. 易于维护。由于三层相互独立,在对中间层进行调整修改时不需要修改客户端的应用程序,这也降低了维护所需要的时间及成本。

3.2　整体架构的实现

整个分布式数据库系统的实现分为四大模块,分别是用户接口模块、系统运行管理控制模块、数据库功能控制模块及写引擎模块。本节接下来会对各个模块进行详细的实现介绍,由于用户接口模块采用的是标准 MySQL 接口模块,文中将不再介绍。

3.2.1　OAM 模块实现

OAM 模块主要实现的是对整个系统的操作管理及维护功能,模块中包含有系统安装运行的脚本、配置文件及功能函数。系统管理功能包括系统管理、模块管理、进程管理等。

3.2.2　数据库控制模块实现

数据库控制模块中包含主要的数据库功能模块,有 DML(Data Manipulation Language)模块、DDL(Data Definition Language)模块、任务模块、执行计划模块等。

(1)DML 模块主要实现的是对操作命令的分析功能,根据操作命令类型分为四个处理模块:InsertPackageProcessor、DeletePackageProcessor、UpdatePackageProcessor、CommandPackageProcessor,这四个处理模块对应着数据操作命令的增、删、改和其余命令。CommandPackageProcessor 模块中处理的命令包括 COMMIT、ROLLBACK、CLEANUP、VIEWTABLELOCK、CLEARTABLELOCK。此外还有 DMLPackageProcessor 模块用于操作检查及索引管理。

(2) DDL 模块主要实现的是对数据库中表和数据操作的处理转译功能,将普通的用户发起的数据库操作请求转化为能够被数据库系统接受并执行的操作请求。

(3) 任务模块中包括了整个分布式数据库操作及管理的具体功能实现,包括系统资源管理、分布式通信引擎、分布式数据的聚合、哈希连接、数据过滤比较、数据元组处理等功能。

(4) 执行计划模块实现了将数据查询任务转换为具体对数据表查询操作的功能,通过对执行计划的优化,可以实现更有效率的查询操作。执行计划模块中子模块包括 Operator、SessionManager、SelectExecutionPlan、ObjectIDManager、AggregateColumn、SelectFilter、ExpressionParser 等功能子模块。

3.2.3　写引擎模块实现

本系统的写引擎模块采用的是 MySQL 数据库的开源存储引擎 infinidb,infinidb 存储引擎是由 Calpont 公司开发的一个在 MySQL 数据库上实现列式存储的存储引擎。本系统的目标应用场景是海量数据的分析处理,列式存储在海量数据压缩存储及查询应用方面具有一定的性能优势,因此选用实现列式存储的 infinidb 存储引擎作为本系统的存储引擎。

4　系统应用性能分析

文中的分布式数据库系统测试硬件环境为 8 台 PC 组成的 Shared-Nothing 架构系统,由带宽为 100Mb/s 的局域网连接。每台 PC 参数为:操作系统:Micrsoft Windows 7 旗舰版 64 位;CPU:Intel(R)Core(TM)i5 M560 2.67GHz;内存:4G。

4.1　基准性能对比结果

文中采用星型模型基准(Star Schema Benchmark)scale 为 1 的大约 1G 字节数据来进行单节点测试,测试查询内容包括星型模型基准中的 Q2.1、Q2.2、Q2.3、Q3.1、Q3.2,作为参照的是 Oracle8g 数据库。测试结果如表 1 所示:

表 1　基准性能测试结果

Query	单节点数据库系统 执行时间	Oracle8g 执行时间
2.1	5.74s	7.46s
2.2	4.52s	4.23s
2.3	3.97s	6.36s
3.1	8.21s	9.46s
3.2	5.58s	8.79s

在基准查询性能方面,具有列存储特性的本系统节点在查询性能方面几乎全面超过了传统行式数据库系统。需要说明的是,虽然查询性能出色,但基于列存储的本系统在表构建数据导入方面性能与传统行式数

据库仍然存在一定差距。

4.2 扩展性能对比结果

扩展性能测试同样采用上节测试中所采用的数据,测试内容为系统在添加节点后星型模型基准中的 Q2.1、Q2.2、Q2.3、Q3.1、Q3.2 查询性能的改善情况。测试结果为表 2:

表 2 扩展性能测试结果

Query	单节点执行时间	双节点执行时间	四节点执行时间	八节点执行时间
2.1	5.74s	2.81s	1.40s	0.75s
2.2	4.52s	2.25s	1.12s	0.59s
2.3	3.97s	1.97s	0.98s	0.51s
3.1	8.21s	4.11s	2.06s	1.06s
3.2	6.58s	3.37s	1.72s	0.89s

扩展性能分析表明,系统查询性能随着节点数目增加可以实现近乎线性的增长。采用基于 Shared-Nothing 架构的本系统具有出色的扩展性能,只需要向系统内添加新的普通节点就能够实现查询性能接近线性的有效提升,这种特性能够帮助数据库系统较好地适应海量数据分析处理场景下查询性能要求飞速提升的情况,系统只需要根据性能要求添加处理节点就可以满足查询性能提升的要求。

5 结束语

文中给出了一种基于 Shared-Nothing 机制的分布式数据库构建方法,通过对实现系统的测试实验,证明了该系统与传统行式数据库相比具有一定的性能优势。由于采用 Shared-Nothing 机制,系统具有出色的可扩展性能。同时列存储特性也有效提升了数据库系统的读取性能。

虽然系统设计的预期功能基本实现,但依然存在很多问题需要进一步研究解决。例如本系统在创建新表时执行时间过长,以及对数据进行修改操作时的执行效率较低等问题。这些问题需在今后的研究工作中进一步加以改进完善。

参考文献:

[1] 马 垣. 关系数据库理论[M]. 北京:清华大学出版社,1999.

[2] 孙风栋,闫海珍. Oracle 10g 数据库系统性能优化与调整[J]. 计算机技术与发展,2009,19(2):82-86.

[3] 夏义全. 数据库应用系统优化方法的研究[J]. 计算机技术与发展,2008,18(7):149-152.

[4] Abadi D J, Madden S R, Hachem N. Column-stores vs row-stores: how different are they really? [C]//Proc. of SIGMOD. [s. l.]:[s. n.],2008.

[5] Mike S, Daniel J A. C-Store: A column-oriented DBMS [C]//Proceedings of 31st International Conference on Very Large Data Bases (VLDB). Trondheim, Norway: Association for Computing Machinery,2005:553-564.

[6] 肖 凌,刘继红,姚建初. 分布式数据库系统的研究与应用[J]. 计算机工程,2001,27(1):33-35.

[7] 曾国林,傅秀芬,吕占德. 异构数据库集成中间件的设计与实现[J]. 计算机技术与发展,2011,21(3):82-86.

[8] 王 君,祝永志,魏榕晖,等. 基于 Oracle 分布式数据库的查询优化[J]. 计算机技术与发展,2008,18(1):157-160.

[9] Abadi D J, Myers D S, DeWitt D J, et al. Materialization strategies in a column-oriented DBMS [C]//23rd International Conference on Data Engineering (ICDE). Istanbul, Turkey: Inst. of Elec. and Elec. Eng. Computer Society,2007:466-475.

[10] Abadi D J. Query execution in column-oriented database systems[D]. USA: MIT,2008.

[11] Abadi D J, Madden S R, Ferreira M. Integrating compression and execution in column-oriented database systems [C]// SIGMOD. [s. l.]:[s. n.],2006:671-682.

[12] Binnig C, Hildenbrand S, Faerber F. Dictionary-based Order-preserving String Compression for Main Memory Column Stores[C]. Providence: ACM,2009.

(上接第 78 页)

[7] Ye Fan, Alvin C, Lu Songwu, et al. Scalable Solution to Minimum Cost Forwarding in Large Sensor Networks[C]//Tenth International Conference on Computer Communications and Networks. [s. l.]:[s. n.],2001:304-309.

[8] Azim M A, Kibria M R, Jamalipour A. An optimized forwarding protocol for lifetime extension of wireless sensor networks[J]. Wireless Communications and Mobile Computing, 2009, 9(1):103-115.

[9] Wang C F, Ding J W, Lee C C. Joint optimization of energy allocation and routing problems in wireless sensor networks[J]. Wireless Communications and Mobile Computing, 2010, 10(2):171-187.

[10] 沈丹丹. 无线传感器网络的节能路由及其优化研究[D]. 杭州:浙江工业大学,2008.

[11] 戴海清,李春茂,陈东发. 无线传感器网络节能策略[J]. 广西师范学院学报,2007,24(2):91-94.

[12] 徐久强,丁玉官,赵 海,等. 无线传感器网络中能量均衡可靠路由度量方法[J]. 计算机工程,2009,35(12):87-89.

一种Shared-Nothing分布式数据库的构建方法

作者： 龙源， 郑彦
作者单位： 南京邮电大学 计算机学院,江苏 南京 210003
刊名： 计算机技术与发展
英文刊名： Computer Technology and Development
年，卷(期)： 2012(10)