

# MongoDB 数据库中 Sharding 技术应用研究

梁海

(桂林电子科技大学 计算机科学与工程学院, 广西 桂林 541004)

**摘要:**非关系型数据库的出现,对于解决面向文档的超大规模和高并发的问题提供了卓有成效的解决方案。MongoDB 为了提高处理大数据量的性能,提供了分片集群的功能,支持自动分片和划分架构,可以利用它构建一个水平扩展的数据库集群系统,将数据库分表存储在各个 Sharding 节点上。文中在研究 MongoDB 特性的基础上,着重分析 Sharding 技术的应用,通过比较普通和分片这两种情况下的性能测试,提出使用 MongoDB 中的 Sharding 技术来解决随着数据量增加带来的数据库的读写性能和效率的问题。

**关键词:**非关系型数据库;性能测试;MongoDB;Sharding

中图分类号:TP39

文献标识码:A

文章编号:1673-629X(2014)07-0060-03

doi:10.3969/j.issn.1673-629X.2014.07.015

## Application and Research on Sharding Technology in MongoDB Database

LIANG Hai

(College of Computer Science and Engineering, Guilin University of Electronic Science and  
Technology, Guilin 541004, China)

**Abstract:** The emergence of non-relational databases, provide the effective solutions for the problems of large scale and high concurrency oriented document. In order to improve the performance of processing large amounts of data, MongoDB provides the ability to patch clusters, supports automatic partitioning and divides a schema, can use it to build a horizontal extension of the database cluster system, the database tables are stored in the Sharding node. Based on research of MongoDB feature, focus on the application of Sharding, through comparing normal and Sharding in both cases of performance testing, present to use the Sharding in MongoDB for solving the problems of database reading and writing performance and efficiency generated by data volumes increasing.

**Key words:** non-relational databases; performance test; MongoDB; Sharding

## 0 引言

MongoDB 是一个基于分布式文件存储的数据库开源项目,是一个基于对介于关系数据库和非关系数据库之间的产品,主要为 Web 应用提供可扩展的高性能数据存储解决方案<sup>[1]</sup>。MongoDB 为了提高处理大数据量的性能,提供了分片集群的功能,Sharding 可以理解为传统数据库表分区的扩展,在数据分片设计时,是从分配的观点来看,根据具有“相同性质”的元组或属性进行分组,使具有“相同性质”的元组或属性划分在一个组中,每组就构成一个片段<sup>[2]</sup>。因此,如果同一个片段的任意两个元素具有“相同性质”的话,那么数

据分配时所用的任意一种方法都将把这两个元素放在一起,以这种方式得到的片段将是分布式数据库中数据合适的分配和存储单位<sup>[3]</sup>。

## 1 MongoDB 介绍

MongoDB 是一个高性能、开源、模式自由的文档型数据库,它在许多场景下可用于替代传统的关系型数据库或键/值存储方式<sup>[4]</sup>。MongoDB 支持的查询语言非常强大,其语法类似于面向对象的查询语言,可以实现类似关系数据库单表查询的绝大部分功能,还支持对数据建立索引。它的特点是高性能、易部署、易使

收稿日期:2013-08-17

修回日期:2013-12-07

网络出版时间:2014-04-24

基金项目:广西壮族自治区学位与研究生教育改革与发展专项课题(201110595R07);中国学位与研究生教育学会信息管理委员会 2011 年课题(2011XX35)

作者简介:梁海(1982-),男,广西北流人,讲师,研究方向为管理信息系统、数据挖掘。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20140424.0812.051.html>

用,支持的数据结构非常松散,支持类似 JSON 的 BSON 格式,存储数据非常方便,可以存储比较复杂的数据类型<sup>[5]</sup>。

BSON 是一个类 JSON 轻量级的二进制数据格式,支持内嵌的文档对象和数组对象,但是 BSON 有 JSON 没有的一些数据类型,如 Date 和 BinData 类型<sup>[6]</sup>。MongoDB 能够使用 BSON,并将 BSON 作为数据的存储存放在磁盘中。当 Client 端要写入文档,使用查询等操作时,需要将文档编码为 BSON 格式,然后再发送给 Server 端。同样,Server 端的返回结果也是编码为 BSON 格式再返回给 Client 端的。使用 BSON 格式的优越性主要体现在以下三个方面<sup>[7]</sup>:

①效率。BSON 是为效率而设计的,只需要使用很少的空间就能实现存储和索引。它的每一个元素的长度存在元素的头部,这样只需要读取到元素长度就能直接寻找到指定的点上进行读取。BSON 格式即使在最坏的情况下,也比 JSON 格式在最好的情况下存储效率高。

②传输性。作为网络数据交换的一种存储形式,BSON 在网络数据交换方面表现出极强的灵活性。在某些情况下,BSON 会牺牲额外的空间让数据的传输更加方便。比如,字符串的传输的前缀会标识字符串的长度,而不是在字符串的末尾打上结束的标记,这样的传输形式有利于 MongoDB 修改传输的数据。

③性能。BSON 格式的编码和解码都是非常快速的,由于使用了 C 语言风格的数据表现形式,在很多语言中均可以便捷地、高效地调用。

在当前的大多数 Web 架构当中,数据库是最难进行横向扩展的,当一个应用系统的用户量和访问量与日俱增的时候,数据库却没有办法像 Web Server 和 APP Server 那样简单的通过添加更多的硬件和服务节点来扩展性能和负载能力<sup>[8]</sup>。特别是对数据库高并发读写、高可扩展性和海量数据的高效率访问这几个方面,传统的关系型数据库难以满足需求,即使可以通过分区、分表和分库等手段来优化,但这种处理代价太高,需要修改较多的逻辑和代码,经常得不偿失<sup>[3]</sup>。但 MongoDB 却很容易解决横向扩展的问题,MongoDB 中集合没有模式,对于存储在 MongoDB 数据库中的文件,不需要知道它的任何结构定义,还可以把不同结构的文件存储在同一个数据库里。

## 2 Sharding 技术

“Shard”作为数据库相关的技术用语,“Sharding”一般称为“分片”。Sharding 不是某个特定数据库软件附属的功能,而是在具体技术细节之上的抽象处理,是水平扩展的解决方案,其主要目的是为突破单节点数

据库服务器的 I/O 能力限制,解决数据库扩展性问题<sup>[9]</sup>。传统的数据库分区的做法是把一大块数据分成了  $n$  小块,查询的时候可以快速定位到某一小块上,在小块中寻址要快很多;由于 CPU 的运算速度远远比磁盘 IO 的速度快,而硬件上又有多个磁盘或者 RAID,可以让数据库驱动 CPU 同时去读写不同的磁盘,这样才有可能提高效率,因此,数据库分区技术用的好的话可以显著地提高性能和效率。

MongoDB 数据库的自动分片技术就是将原先数据库中集合依据一定的规则切分成若干小块,这些分片小块统一由 mongos 路由管理,当有请求查询或写入时,路由会依据分片 shard key 规则找到对应的分片操作。分片解决了写密集操作,用于分散单一写服务器负载。亦或者原先的存储空间不够了,这个时候可能通过分片操作将之后的数据写入其他存储空间上。可以看出,集合的分片和数据库的分表类似,并且每个分片都支持写操作<sup>[10]</sup>。当各 Sharding 间负载和数据分布不平衡时自动恢复平衡,能够简单方便地添加、删除节点,并且具备系统自动故障转移的功能,还可以扩展至上千台节点。

Sharding 的思想是从分区的思想而来,但数据库分区基本上是数据对象级别的处理,比如表和索引的分区,每个子数据集上能够有不同的物理存储属性,还是单个数据库范围内的操作,而 Sharding 是能够跨数据库,甚至跨越物理机器的<sup>[11-12]</sup>。Sharding 与数据库分区的区别如表 1 所示<sup>[13-14]</sup>。

表 1 Sharding 与数据库分区的区别

	Sharding	分区
存储依赖	可跨越数据库,可跨越物理机器	可跨越表空间,不同的物理属性,不能跨数据库存储
数据划分	常见为时间、范围、面向服务等	范围、Hash、列表、混合分区等
储存方式	分布式	集中式
扩展性	Scale Out	Scale Up
可用性	无单点	存在单点
成本	低廉	适中甚至昂贵
应用场合	常见于 Web 2.0 网站、政府机关网站、金融机构管理系统等	多数传统场合

在 MongoDB 里面,Sharding 一般有以下几种情况:

①一个或多个分片,每个分片承载全部数据的一个分区。读和写被自动发送到合适的分片,每个分片都由一个备份集支持。一个备份集就是一个或多个服务器,每个都承载着相同数据的拷贝,其中一个主要是集而其他都是次要集。若主要集失效则其中一个次要集自动变为主要集,所有的写入和一致读出都在主要集上,所有的最终一致读出都分配在次要集中。

②多个配置服务器,每个都承载着元数据的一个

拷贝,用以表明哪些数据在哪个分片上。

③一个或多个路由器,每个都为—个或多个客户端充当一个服务器。客户端向路由器发送查询更新请求,路由器在查询配置服务器后将它们路由给合适的分片。

④一个或多个客户端,每个都是用户应用(的一部分),并且通过 mongo 客户端库(驱动)自己的语言向路由器发送命令。

设置分片时,需要从集合里面选一个键,用该键的值作为数据拆分的依据。这个键成为片键 shard key,值必须是不可更改的。一旦这个值被更改,就有可能从一个节点被挪到另一个节点,从而带来很大的开销。shard key 要有一定的随机性而不是单向增长。单向增长的 shard key 会导致新插入的数据都位于一个 chunk 中,在某一个 shard 节点中产生集中的写压力。因此应尽量避免直接使用\_id、时间戳等这种单向增长的值作为 shard key。

### 3 性能测试

此次测试共涉及 3 台服务器:1 台 Web 服务器,2 台 MongoDB 服务器。机器配置:CPU 为 Intel(R) Core (TM) 2 Duo CPU T6570 @ 2.10 GHz、内存为 16 G DDR2 667、硬盘为 SATA 1T、操作系统为 Windows Server 2008 R2。分别在 2 台机器运行一个 mongod 实例和一个 mongs 进程。使用 C#语言编写程序进行 MongoDB 千万级数据量的性能测试,分别测试如下几个项目:

#### 1) 普通插入性能。

如图 1 所示,在插入的数据每条为 1 kB 的情况下,普通插入的方式在数据量小于 1 000 万条时,性能是比较高效的,但之后急剧降低,通过查看任务管理器,发现此时的内存占用率高于 90% 以上,说明在数据的插入跟内存有着直接的关系,数据量过大时,需要在磁盘和内存间进行大量的数据交换,因此性能下降较快。而在分片的情况下,则相对稳定。

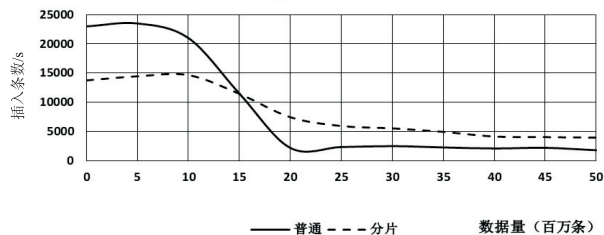


图 1 普通插入性能比较图

#### 2) 批量插入性能。

如图 2 所示,使用批量操作 InsertBatch,主要测试批量插入性能的提高程度,发现跟普通插入差别较小,原因是网络带宽的限制。分片比起普通的,有一定的

优势。

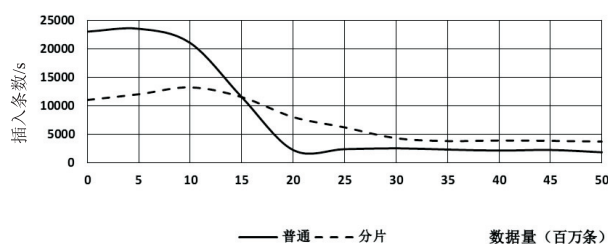


图 2 批量插入性能比较图

#### 3) 安全插入功能。

如图 3 所示,在安全模式下确保数据都能成功写入数据库,相比普通插入和批量插入的差距,体现出相对稳定的特点。而分片体现出来的稳定性能明显优于普通模式。

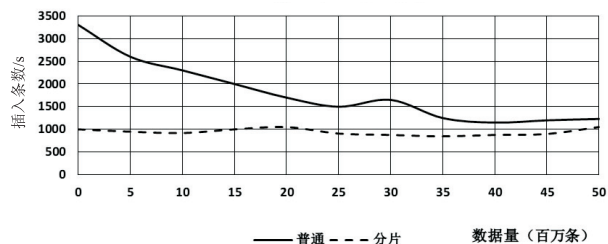


图 3 安全插入性能比较图

#### 4) 查询 1 000 条记录。

如图 4 所示,在没有设置条件的情况下,查询 1 000 条记录的时候,在分片的情况下,性能表现比较稳定。

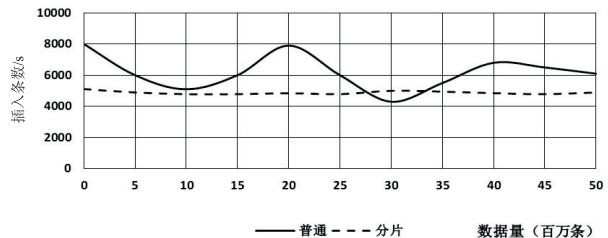


图 4 查询性能比较图

## 4 结束语

经过测试分析得知,在数据量较小时并且数据频繁交换的情况下使用竖向分片,可以保证程序设计中逻辑相对保持简单,提高了程序的可靠性,降低了程序员的负担。当竖向分片仍然不足以解决问题的情况下,再引入横向分片,将数据经常交换的表分为一组,横向分片数要尽量少,减少统计程序的复杂性。在要求更高效率的情况下,横向分片加并发的数据检索可大大提高数据统计速度,可以满足实时高速的要求。

## 参考文献:

[1] Banker K. MongoDB 实战[M]. 丁雪丰,译. 北京:人民邮电

(下转第 67 页)

合推理,李雷选择运动模式,语义系统自动生成适合用户的方案。

智能家居系统		
当前居家模式: 运动模式		
用户基本信息:		当前设备状态:
姓名: 李雷	空调:	开启状态: 开启
性别: 男		温度: 26℃
年龄: 30岁	热水:	开启状态: 开启
室温偏好: 28℃		温度: 45℃
水温偏好: 45℃	新风:	开启状态: 开启
湿度偏好: 45RH		温度: 60RH
<div>更改设置</div>		

图 2 功能界面显示

如果用户对当前设置不满意,可以点击更改设置按钮进行设置满足需求。

5 结束语

文中在分析智能家居人工智能化的需求和现状之后,指出了语义技术在智能家居中的应用前景;在研究了本体与语义推理、X 列表相关理论的基础上,建立了一种智能家居语义人工智能控制模型,并实现了系统的关键功能。系统在实际应用中证明了语义技术应用在智能家居系统中可以进一步提升以“用户为中心”的智能化需求。

参考文献:

[1] 冯凯,童世华.智能家居的由来及其发展趋势[J].中国新技术新产品,2010(6):7-7.

[2] 杨华杰,张尧学,周悦芝,等.基于 Agent 实现 Web 服务的

(上接第 62 页)

出版社,2012.

[2] MongoDB manual[EB/OL]. 2012. <http://docs.mongodb.org/manual/sharding/>.

[3] Chodorow K. Scaling MongoDB[M]. [s.l.]:O'Reilly Media, 2011.

[4] Tudorica B G,Bucur C. A comparison between several NoSQL databases with comments and notes[C]//Proceedings of the 10th Roedunet international conference. Lasi:IEEE,2011.

[5] 张文盛,郑汉华.基于 MongoDB 构建高性能网站技术研究[J].吉林师范大学学报(自然科学版),2013(1):123-127.

[6] Chodorow K,Dirolf M. MongoDB 权威指南[M].程显峰,译.北京:人民邮电出版社,2011.

[7] Weth C,Datta A. GutenTag: A multi-term caching optimized TagQuery processor for key-value based NoSQL storage sys-

按需计算[J]. 计算机工程与应用,2005,41(19):132-136.

[3] European Commission. Internet of Things in 2020-A Roadmap for the Future[R]. [s.l.]:[s.n.],2008.

[4] 蒋鹏,王波.多 Agent 系统技术在智能建筑中的应用[J].建筑电气,2005,24(5):41-44.

[5] Wang W,Sesmero J V. Report on reference architecture IoT service creation and provision[R]. [s.l.]:[s.n.],2012.

[6] Barnaghi P,Wang W,Henson C,et al. Semantics for the internet of things:early progress and back to the future[J]. International Journal on Semantic Web and Information Systems, 2012,8(1):1-21.

[7] 陈莉.基于领域本体的智能搜索系统的研究和应用[D].南京:南京航空航天大学,2008.

[8] Koshizuka N,Sakamura K. Ubiquitous ID: standards for ubiquitous computing and the internet of things[J]. Pervasive Computing,2010,9(4):98-101.

[9] Euzenat J,Valtchev P. An integrative proximity measure for ontology alignment[C]//Proceedings of the semantic integration workshop at the international semantic web conference. [s.l.]:[s.n.],2003:202-205.

[10] Studer R,Benjamins V R,Fensel D. Knowledge engineering, principles and methods[J]. Data and Knowledge Engineering, 1998,25(1-2):161-197.

[11] 杜小勇,李曼,王大治.语义 Web 与本体研究综述[J]. 计算机应用,2004,24(10):14-16.

[12] 李从东,谢天,汤勇力,等.面向云制造服务的语义 X 列表知识表达与推理体系[J]. 计算机集成制造系统,2012, 18(7):1469-1484.

[13] 宋伟,张铭.语义网简明教程[M].北京:高等教育出版社,2008.

[14] 向阳,王敏,马强.基于 Jena 的本体构建方法研究[J]. 计算机工程,2007,33(14):59-61.

tems[EB/OL]. 2011. <http://arxiv.org/pdf/1105.4452>.

[8] 姚林,张永库. NoSQL 的分布式存储与扩展解决方法[J]. 计算机工程,2012,38(6):40-42.

[9] 潘凡.从 MySQL 到 MongoDB-视觉中国的 NoSQL 之路[J]. 程序员,2010(6):79-81.

[10] 毕洪宇.利用 NoSQL 构建高性能全文检索系统[J]. 计算机与现代化,2012(3):122-124.

[11] 刘艳俊,敖杰刚,徐齐行.基于 MongoDB 云计算下 GML 分布式集群环境搭建研究[J]. 测绘标准化,2012(1):3-5.

[12] 卞静,严益强.用对象分片方法优化数据库设计[J]. 计算机应用,2004,24(z1):118-119.

[13] 何杭锋.基于 FODO 算法 MongoDB 自动分片的改进[J]. 计算机技术与发展,2013,23(7):127-130.

[14] 冯大辉.开源数据库 Sharding 技术[J]. 程序员,2008(7): 92-93.



作者: 梁海, LIANG Hai  
作者单位: 桂林电子科技大学 计算机科学与工程学院, 广西 桂林, 541004  
刊名: 计算机技术与发展   
英文刊名: Computer Technology and Development  
年, 卷(期): 2014(7)

参考文献(14条)

1. Banker K;丁雪丰 MongoDB实战 2012
2. MongoDB manual 2012
3. Chodorow K Scaling MongoDB 2011
4. Tudorica B G;Bucur C A comparison between several NoSQL databases with comments and notes 2011
5. 张文盛;郑汉华 基于MongoDB构建高性能网站技术研究 2013(01)
6. Chodorow K;Dirolf M;程显峰 MongoDB权威指南 2011
7. Weth C;Datta A GutenTag:Amulti-term caching optimized TagQuery processor for key-value based NoSQL storage sys-tems 2011
8. 姚林;张永库 NoSQL 的分布式存储与扩展解决方法 2012(06)
9. 潘凡 从MySQL到MongoDB-视觉中国的NoSQL之路 2010(06)
10. 毕洪宇 利用NoSQL构建高性能全文检索系统 2012(03)
11. 刘艳俊;敖杰刚;徐齐行 基于MongoDB云计算下GML分布式集群环境搭建研究 2012(01)
12. 卞静;严益强 用对象分片方法优化数据库设计 2004(z1)
13. 何杭锋 基于FODO算法MongoDB自动分片的改进 2013(07)
14. 冯大辉 开源数据库Sharding技术 2008(07)

引用本文格式: 梁海. LIANG Hai MongoDB数据库中Sharding技术应用研究[期刊论文]-计算机技术与发展 2014(7)