基于 Postgres-XL 的数据管理优化技术研究

艾丽蓉,李 凯
(西北工业大学 计算机学院,陕西 西安 710129)

摘 要:面对城市中快速增长的数据量,单机数据库的性能明显应对不足。针对单机数据性能低下、可扩展性差以及数据库安全性等问题,利用开源分布式数据库对数据进行管理显得十分必要。Postgres-XL 作为一个完全满足 ACID 的、开源的、可方便进行水平扩展的、多租户安全的数据库解决方案,可以有效地完成数据管理工作。为了进一步提升 Postgres-XL 的效率,对 Postgres-XL 的扩展模块和存储管理模块进行了研究。首先提出利用 Postgres-XL 的表继承等特性完成表拆分,然后利用基于 SQL/MED 的中间件把 Redis 作为 Postgres-XL 的缓存,最后在 Postgres-XL 外存管理模块中增加 SSD 缓存模块。实验结果表明,以上三种优化策略有效地降低了数据库请求的时间延迟,提升了 Postgres-XL 的效率。

关键词:Postgres-XL;二级缓存;分区表;Redis;SSD
中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2018)03-0011-04
doi:10.3969/j.issn.1673-629X.2018.03.003

Study on Optimization Technology of Data Management Based on Postgres-XL

AI Li-rong,LI Kai
(School of Computer Science,Northwestern Polytechnical University,Xi'an 710129,China)

Abstract:As the amount of data in the city grows rapidly,the performance of stand-alone database is clearly deficient. In order to solve problems of low performance,poor scalability and the security of database,it is necessary to use the open-source distributed database in data management. As an open source and horizontally-scalable database that meets the requirement of ACID and is secure for multiple tenancies,Postgres-XL can efficiently complete the data management. In order to further improve the efficiency of Postgres-XL,the expansion and storage management modules of Postgres-XL are studied. Firstly we propose to utilize the table inheritance and other characteristics of Postgres-XL to complete the table split,then use Redis as the cache of Postgres-XL based on SQL/MED middleware,and increase SSD cache modules in the external memory management module of Postgres-XL in the end. Experiments show that proposed optimization strategies can effectively reduce the latency of the database request and improve the processing efficiency of Postgres-XL.

Key words:Postgres-XL;secondary cache;partition table;Redis;SSD

0 引 言

面对日益增长的海量数据,传统的单机集中式数据库的弊端日益显现[1-2]。为了解决单机数据库下数据存不下、难于扩展、查询延迟大等问题,需要在横向或纵向上升级数据库配置。在纵向升级配置提高数据库的性能主要是采用大型机、大型机与 PGStorm 的方式。其中 PGStorm 可将 CPU 的密集型工作负载转移到 GPU 处理,从而利用 GPU 强大的并行执行能力完成处理任务。Postgres-XL 是在横向上对 PostgreSQL 进行了扩展。虽然,Postgres-XL 相对 PostgreSQL 提升了数据库处理性能,但是,针对 Postgres-XL 本身的优化也是十分必要的。

1 Postgres-XL 分析

Postgres-XL 是基于 PostgreSQL 数据库实现的真正意义上的分布式数据库。目前大多数数据库的水平拆分的方案都有很多限制,譬如不能跨机器 Join、对 SQL 有各种限制。Postgres-XL 实现得更加彻底,用户访问集群数据库就跟单机数据库一样。

1.1 Postgres-XL 简介

Postgres-XL 全称为 Postgres eXtensible Lattice,它使用了开源协议允许将开源代码与闭源代码混在一起

收稿日期:2017-02-26 **修回日期**:2017-06-28 **网络出版时间**:2017-12-04
基金项目:国家自然科学基金(61502371)
作者简介:艾丽蓉(1970-),女,博士,副教授,研究方向为人工智能;李 凯(1991-),男,硕士,研究方向为数据库、人工智能。

使用。同时它还有自己很多独特的特性,譬如真正完全支持集群级别的数据一致性(ACID)、支持 OLAP 应用、采用 MPP(massively parallel processing,大规模并行处理系统)架构模式、读写可拓展、可用作分布式 Key-Value 存储、支持分布式多版本控制(MVCC)等。

在性能方面,随着数据节点数的增长,Postgres-XL 扩展能力的增加呈线性增长趋势。实际测试结果是,当有 3 个数据节点时可以达到 PostgreSQL 数据库大约 2 倍的性能,5 个节点为 3 倍,10 个节点为 6 倍左右。

1.2 Postgres-XL 分布式架构

Postgres-XL 是一个基于 PostgreSQL 数据库的横向扩展开源数据库集群,拥有足够的灵活性来处理不同的工作负载。

Postgres-XL 支持 MPP,允许数据节点间通信,交换复杂跨节点关联查询相关数据信息。Postgres-XL 的主要节点有:全局事务管理器(global transaction manager,GTM)、协调器(coordinator)、数据节点(datanode)。GTM 主要负责处理多版本并发控制任务,包括数据库事务的 ID 以及数据快照,同时它还管理集群中全局性数据,譬如时间戳等。coordinator 负责处理客户端的网络连接,分析查询语句,生成语句的执行计划,然后将计划发送到目的数据节点执行。数据节点主要存放具体的物理数据。除了上述节点之外,Postgres-XL 为了减轻 GTM 负载,还可以部署 GTM Proxy 节点,GTM Proxy 代理 coordinator 和 datanode 对 GTM 进行访问。

2 优化技术

对 Postgres-XL 的优化主要体现为在有限的时间内提高系统吞吐量。数据库系统吞吐量是指单位时间内处理的事务数。为了提高数据库系统的吞吐量,一般需要提高缓存命中率、降低磁盘 IO。

2.1 数据表优化

随着数据库中数据量的增加,数据库表的查询效率非常低下。为了解决上述问题,基于 Postgres-XL 分区表和表继承特性实现了表的水平拆分以及垂直拆分。Postgres-XL 数据库表的水平拆分是按照某种规则将数据库表进行划分,然后将数据存储到多个结构相同的表中。例如,对于重庆某区的执法系统案件表,可以按照案件处理的时间拆分,2015 年和 2016 年的数据分别存放在结构相同的表中,这样就可以根据不同的时间进行区分,提高访问效率。数据库表的垂直拆分很容易地解决了表跟表之间的 IO 竞争,但是没有解决单个数据表数据量大的问题。而数据库表的水平拆分,解决了大数据表数据量大的问题,但是没有解决表之间的 IO 争夺。因此大多方案是采取两者结合的方式,如图 1 所示。

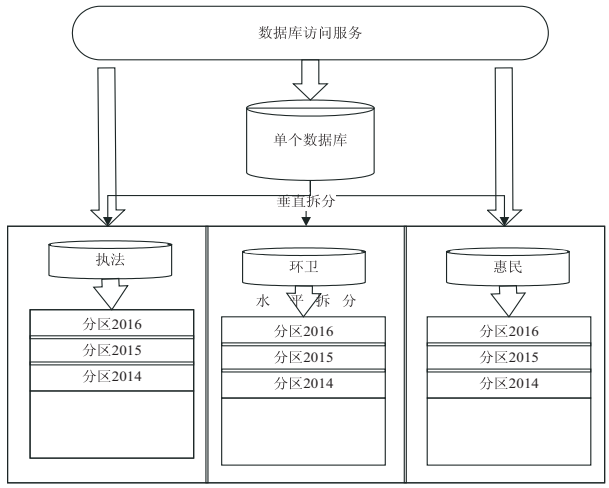


图 1 组合拆分

在传统数据库中,对数据库表进行水平或者垂直拆分的复杂度是非常高的,而且容易产生附属问题,譬如事务、跨表 Join 操作等。目前 Postgres-XL 支持的分区有范围分区和列表分区。范围分区是表根据一个或者多个字段拆分成范围,在不同范围内存放没有键值重复的数据;列表分区指在表中明确指出每个分区里应该出现哪些字段[3]。

2.2 基于 Redis 的优化

SQL/MED(management of external data)是 SQL 语言中管理外部数据的一个扩展标准,这个标准定义在 SQL:2003 中,它通过定义一个外部数据包装器和数据连接类型来管理外部数据[4]。Postgres-XL 提供了对 SQL/MED 标准的支持,通过 SQL/MED 连接到各种异构数据库。

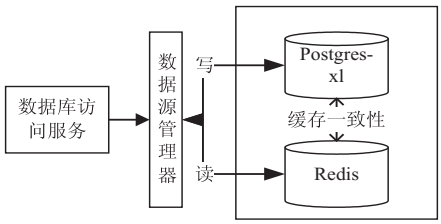
基于 SQL/MED 标准设计实现从 Postgres-XL 访问 Redis 数据库的中间件,将 Redis 中的数据映射到 Postgres-XL 中[5-6]。传统的读写分离方案要在数据访问端进行数据缓存控制,需要控制多个数据源,复杂度很高。通过基于 SQL/MED 中间件把 Redis 作为 Postgres-XL 伪组件进而统一数据源,同时提高了其处理速度。读写架构如图 2 所示。

数据库访问服务发送数据请求,如果是写操作则直接操作 Postgres-XL 同时更新 Redis 缓存数据。对于读操作,则由 Postgres-XL 直接读取 Redis 返回数据。根据 Postgres-XL 的规则特性,利用中间件对 Redis 进行更新。

2.3 基于 SSD 的优化

目前使用 SSD 对 Postgres-XL 优化的策略分为三种。第一种是把 Postgres-XL 所有节点全部安装在 SSD 上,即把传统机械磁盘全部换成 SSD。这种方式简单,但是 SSD 的擦写次数有限,分布式集群机器众

多,因此成本非常高。第二种是将非关键数据放到 SSD 上,譬如日志、索引等。第三种是将 SSD 作为 Postgres-XL 的二级缓存来减缓访问机械硬盘的频率[7-8]。下面对第二、三种进行详细的论述。



无SQL/MED时的架构

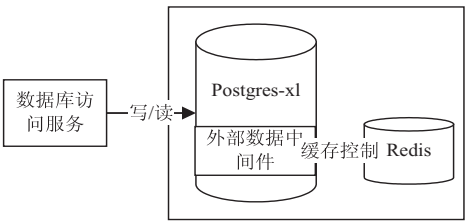


图 2　SQL/MED 架构

2.3.1　非关键性数据存储

在 Postgres-XL 中,将数据操作的日志、索引、临时表放到 SSD 中以提高数据库的整体执行效率,而具体数据存储到机械硬盘上。临时表和索引的特点是顺序写、随机读,而且这些数据属于非关键性且可重新建立。因此,SSD 的损坏不会影响数据库的一致性。

此处利用 Postgres-XL 数据表空间特性将索引与数据库表分开存储,同时把临时表存放在 SSD 上。在 Postgres-XL 中,数据表空间可以为表指定特定存储目录。创建表和索引可以指定表空间,这样表、索引就可以存储到特定的目录中。在 Postgres-XL 中,创建数据表空间语法如下:

Create tablespace TABLESPACE_NAME [OWNER user_name] LOCATION 'directory';

其中 directory 指定是 SSD 挂载的路径。创建好表空间就可以将索引存放在挂载 SSD 的目录中,表存放在机械硬盘中。此时创建测试表 ssd_test,将创建在此表上的索引存到指定表空间,命令如下:

Create index tablespace_index on ssd_test(id)

Tablespace space_test

2.3.2　缓存优化

Postgres-XL 外存管理是由 SMGR 提供对外存操作的统一接口,结构如图 3 所示。SMGR 负责统管各种介质管理器,根据上层的请求选择一个具体的介质管理器进行操作[9-11]。

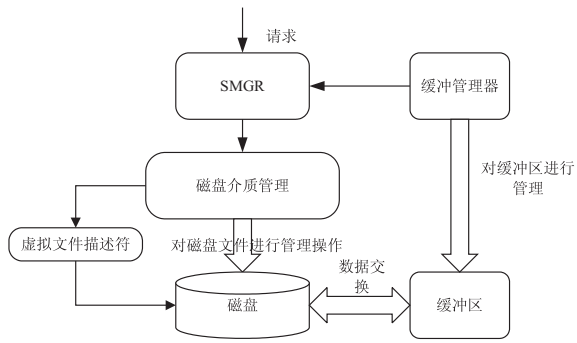


图 3　外存管理系统结构

将 SSD 作为二级缓存的基本思想是在数据节点共享缓存满时,共享缓存发生了页置换。此时并不是将置换出的缓存块刷新到机械硬盘,而是将数据转移到 SSD 中以达到缓存的目的。当下次请求当前数据块时,在 SSD 缓存中将数据返回而非磁盘。处理的基本流程如图 4 所示。

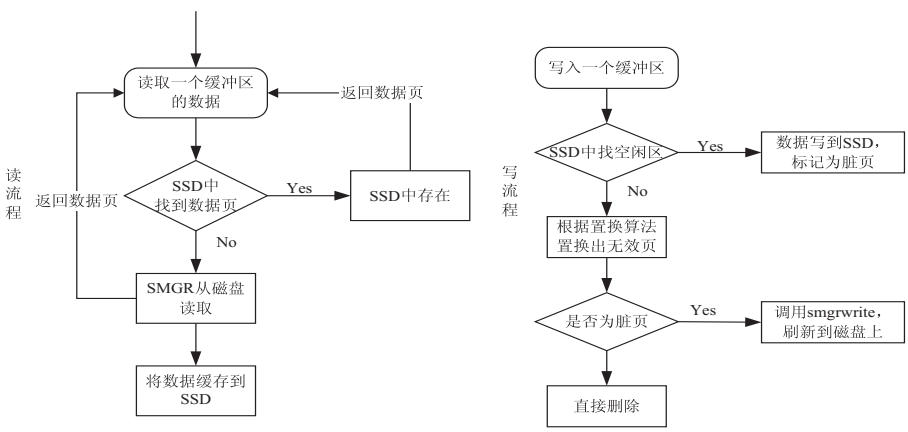


图 4　拥有二级缓存的读写流程

因为成本的问题,混合存储系统中的 SSD 的容量要小于机械硬盘,大约为整个储存空间的 5% ~10%,这其中还包括 SSD 垃圾回收的空间[12-15]。

此处对缓存替换策略实现在原 LRU 算法中加入了 midpoint 而有数据新读取的缓冲块,虽然是最新访问的页,但并不直接放到 LRU 列表首部,而是放入到 LRU 列表的 midpoint 位置。将 midpoint 位置之后的列表称为 old 列表,之前的列表称为 new 列表。这样可以避免某些 SQL 操作将缓冲池中的块全部刷新,譬如数据表扫描操作。对于 old 表中的数据,如果是热

点数据很快会再次被访问到,然后进入到 new 列表。

3 实验结果分析

利用 Postgres-XL 自带的 pgbench 工具来完成实验数据的模拟,然后完成数据库并发访问的模拟,检测上述三种优化策略下的数据库吞吐量。在实验环境中 Postgres-XL 在并发量达到 60 时,数据库性能达到了最高。因此,以下实验并发均为 60,测试时间为 12 小时。实验结果如下:

(1)在表分区的实验中数据量为 10 G,对于 insert 操作,数据表分区前的插入性能强于数据表分区后。随着并发量增大和数据量增大,插入性能相差无几。究其原因,数据表分区时指定了数据的流向规则,耗费了一定的时间。对于 select 和 update 操作,数据表分区后的性能是分区之前的 2.8 倍左右。

(2)在基于 Redis 作为二级缓存方案中,对单数据表进行了测试,采取表缓存后的数据库性能为未采取策略的 6 倍左右。

(3)在基于 SSD 缓存优化方案中,采取 SSD 缓存为硬盘容量的 10%。在上述测试中采用纯 SSD 作为存储介质时,Postgres-XL 的性能是采用纯机械盘的 8 倍左右。采取将非数据节点、相关表的索引放到 SSD 的策略时,Postgres-XL 的性能也在纯机械盘的基础上提升了 2.6 倍。将 SSD 作为二级缓存时,Postgres-XL 的性能提升了大约 5.6 倍。

4 结束语

针对目前大数量单机数据库性能低下产生大量的磁盘 IO 等问题,基于分布式数据库 Postgres-XL,提出了相应的解决手段。通过分区表、Redis、SSD 对 Postgres-XL 进行优化,有效地提升了数据库的处理效率,对于分布式数据库 Postgres-XL 的应用以及数据管理的发展起到了很好的推动作用。

参考文献:

[1] 齐 磊. 大数据分析场景下分布式数据库技术的应用[J]. 移动通信,2015(12):58-62.

[2] 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.

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

[4] 林河水,程 伟,孙玉芳. PostgreSQL 存储管理机制研究[J]. 计算机科学,2004,31(12):76-80.

[5] 郑小裕. SQL 与 NoSQL 数据库的统一查询模型的研究与实现[D]. 长沙:湖南大学,2014.

[6] 白 鑫. 基于 Redis 的信息存储优化技术研究与应用[D]. 北京:北方工业大学,2014.

[7] YANG Q, REN J. I-CASH: intelligently coupled array of SSD and HDD[C]//Proceedings of the 2011 IEEE 17th international symposium on high performance computer architecture. Washington, DC, USA: IEEE Computer Society, 2011:278-289.

[8] 闫 林. 基于 SSD 的 HDD 缓存系统研究[D]. 西安:西安电子科技大学,2014.

[9] LEE R,ZHOU M. Extending PostgreSQL to support distributed/heterogeneous query processing[C]//Proceedings of the 12th international conference on database systems for advanced applications. Berlin: Springer-Verlag, 2007: 1086-1097.

[10] NOGUCHI Y, UMENO H. The design and implementation of multiple buffer cache in PostgreSQL[C]//International conference on control, automation and systems. [s. l.]: IEEE, 2008:2654-2657.

[11] LIU S, UMENO H. Implementation and improvement of TPM for PostgreSQL in Linux[C]//International conference on control, automation and systems. [s. l.]: IEEE, 2008: 2658-2661.

[12] 宋 磊,王静文. PostgreSQL 数据库性能优化[J]. 电脑编程技巧与维护,2009(16):63-66.

[13] 鲁 笛,向 阳,刘增宝. PostgreSQL 数据库缓冲管理的分析与研究[J]. 计算机技术与发展,2011,21(12):41-44.

[14] 刘圣卓,姜进磊,杨广文. 一种面向 SSD-HDD 混合存储的热区跟踪替换算法[J]. 小型微型计算机系统,2012,33(10):2255-2258.

[15] 卢朝霞,习 捷,王 剑. 基于数据库分区的海量数据存储技术的研究[C]//2006 中国控制与决策学术年会. 出版地不详:出版者不详,2006.