计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2019,28(6):69−75 [doi: 10.15888/j.cnki.csa.006936]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于 K-means 的 SAMP 系统数据库查询性能优化策略①

马　跃, 王喆峰, 尹震宇, 王春晓, 李明时, 廉梦佳

[1](中国科学院大学 计算机与控制学院, 北京 100049)
[2](中国科学院 沈阳计算技术研究所, 沈阳 110168)
通讯作者: 尹震宇, E-mail: 704419968@qq.com

摘　要: 中科院仪器设备共享管理平台 (以下简称为 SAMP) 系统有效解决了各科研单位间仪器设备管理封闭、共享困难和运行效率低的棘手问题. 同时, 可以及时了解各类仪器的使用情况、共享情况, 为各级业务主管部门展开科学高效的管理工作提供良好的决策依据. 当 SAMP 系统应用数据库中存储的数据量达到百万级时, 对数据库中预约表和用户表 (或仪器表) 进行连接查询时, 将导致数据表查询性能的下降, 从而影响整个 SAMP 系统的性能. 目前主流的解决方案是采用 Hash 取模算法对数据表进行水平切分, 但预约表中的主键为自动递增的整数, 并没有实际意义, 所以优化效果不理想. 由于预约的用户和被预约的仪器在地理区域上呈现一定的聚集性, 因此本文提出了一种基于 K-means 聚类算法的分表策略, 采用该策略能够将预约表的查询性能提升至少 70%.

关键词: SAMP 系统; 数据库切分; K-means 聚类算法; 查询性能; 并发

引用格式: 马跃,王喆峰,尹震宇,王春晓,李明时,廉梦佳.基于 K-means 的 SAMP 系统数据库查询性能优化策略.计算机系统应用,2019,28(6):69–75. http://www.c-s-a.org.cn/1003-3254/6936.html

Optimal Strategy of Query Performance Based on K-means for SAMP System

MA Yue, WANG Zhe-Feng, YIN Zhen-Yu, WANG Chun-Xiao, LI Ming-Shi, LIAN Meng-Jia

[1](School of Computer and Control Engineering, Universiy of Chinese Academy of Sciences, Beijing 100049, China)
[2](Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang 110168, China)

Abstract: Instrument and equipment sharing management platform for Chinese Academy of Sciences can solve the problems of closed management, difficult sharing, and low operation efficiency of instruments and equipment among scientific units effectively. Meanwhile, users can learn the using and the sharing situation of various instruments through the system. The SAMP system can provide decent decision-making basis for scientific and efficient management work of competent business departments at all levels. So when the data, which belong to the apparatus apply info table, reaches the million scale, the query performance will decline quickly because of using joining query. At present, the solution is using sharding, just like Hash fetching algorithm. Because of the meaningless ID, this way is ineffective. There is a certain degree of aggregation in geographical areas between users and instruments, so a strategy that based on the K-means algorithm is used in this study. The result shows it can improve the query performance at least 70%.

Key words: SAMP system; sharding; K-means algorithm; query performance; concurrency

目前 SAMP 系统以划分的 14 个区域中心为逻辑层级结构，在物理结构设计上采用集中的单系统架构, 这使得系统的架构层次变得简洁.同时随着系统间与模块间的交互变得可控, 也减轻了系统交换层的压力. 但

① 基金项目: 核高基重大专项 (2017ZX01030-201)
Foundation item: National Science and Technology Major Program (2017ZX01030-201)
收稿时间: 2018-12-10; 修改时间: 2018-12-29; 采用时间: 2019-01-10; csa 在线出版时间: 2019-05-25

由于 SAMP 系统已在全院 114 个所运行, 8000 台设备在线，因此这种架构设计在应对每天千级以上的并发访问及百万级以上数据存储问题时, 系统整体性能有待提高.

1 SAMP 系统性能瓶颈分析

1.1 SAMP 系统实体关系概念模型

系统中存在着两个核心的实体, 分别为用户和仪器. 系统中的其他实体, 如研究所、仪器管理员、委托单、耗材、实验标准等均是在用户与仪器的基础上抽象衍生出来的.

在实体之间还存在着多种关系, 这种关系可能是实体间的包含关系, 也可能是实体间的操作关系. 例如研究所和仪器之间具有"属于"的包含关系, 用户和仪器之间具有包括"使用"、"预约"、"查看"或者"收藏"等多种操作关系. 根据上面的描述, SAMP 系统实体关系核心概念模型如图 1 所示.

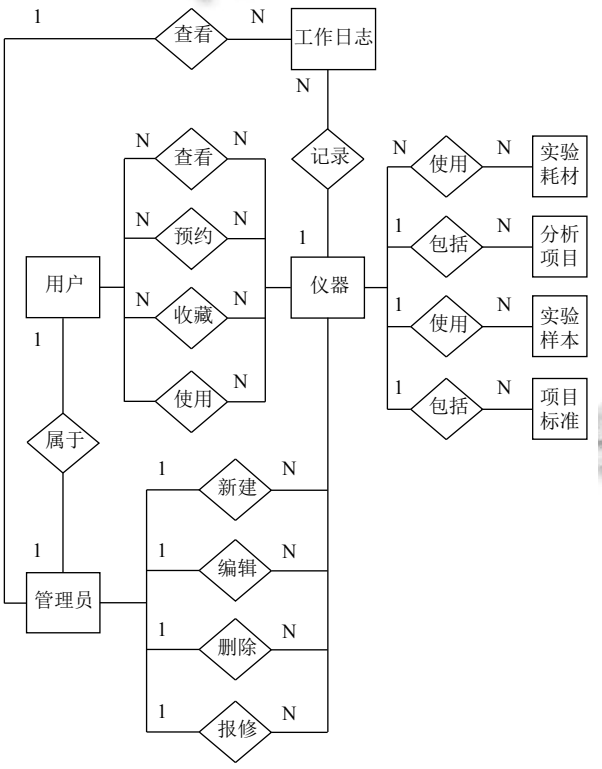

图 1 SAMP 系统实体关系核心概念模型图

1.2 SAMP 系统性能潜在瓶颈

SAMP 系统的核心业务之一为用户对仪器的预约. 由于用户对仪器的预约操作关系在 SAMP 系统实体关系概念模型中是一个多对多的关系, 因此在数据库中需要单独建表处理该关系, 即创建预约表. 该表中记录了用户的全部预约仪器记录, 这样导致了以下两个问题:

(1) 随着请求并发量的提高, 由于系统在处理预约操作的时需要对数据库进行并发场景下的读写操作, 千级并发量导致系统整体性能明显下降.

(2) 当预约表数据量达到百万级别时, 对预约表进行查询操作将消耗大量系统资源与时间.

由此可见, 用户对仪器的预约操作以及后续对该预约委托单状态的维护将使应用服务器与数据库之间发生频繁的读写操作, 因此用户对仪器的预约操作使得系统整体性能受到影响.

1.3 待解决的核心问题

从数据库层面去考虑处理高并发与海量数据存储问题的基本的解决方案[1], 国内外现有的研究普遍从如下数据存储结构与数据查询性能两个方面出发提出一系列解决方案. 例如文献[2]中作者提出了一种基于哈希算法的数据库切分 (sharding) 策略, 并试图从数据存储结构方面解决海量数据存储问题并减少数据库的负担, 缩短大数据量表的查询时间, 从而达到提升系统整体性能的目的. 文献[3]中作者提出了通过在 MySQL 数据库中合理地设置索引与游标的方式达到提高数据表的查询效率. 而在这之前, 还需要解决如下问题[4–7]:

(1) 确定切分规则, 即采用何种策略能够将数据合理的切分开, 使得在后续从不同维度对预约表进行查询时都能保证查询性能. 预约表与用户表和仪器表之间存在主外键关系, 因此预约表中的主键为自动递增的整数, 并没有实际意义. 现有研究中通常采用的 Hash 取模切分算法将不再适用, 必须根据预约表的特点制定合理的切分规则[8–11].

(2) 对数据库进行切分后, 处理查询请求时必须准确定位该请求需要查询的子库或子表.

(3) 当各个子库和子表的容量达到阈值时采取何种策略进行扩容.

根据目前 SAMP 系统中对区域中心的定义和划分以及系统架构中的各个所级服务器与中心服务器之间的关系, 可以看出用户对仪器的预约操作存在一定的聚集性. 因此, 核心问题被定义为: 选取合适的特征变量对预约记录进行聚类分析, 并根据聚类结果选取合理的切分规则对数据库进行水平切分, 使得切分后的数据库能够提升预约表的查询性能, 从而提高整体系统的性能[12–15].

2 基于聚类的数据库分表

根据 SAMP 系统实体关系概念模型可知, 对预约表进行聚类分表时特征变量的选择需要结合用户表与仪器表共同分析. 用户表、仪器表以及预约表的核心表结构如表 1、表 2 和表 3 所示.

表 1 用户表核心表结构

字段名	字段类型	字段释义
userID	int(11)	用户 ID
userName	varchar(64)	用户名
belongAreaCenterID	int(11)	所属区域中心 ID
belongInstituteID	int(11)	所属研究院 ID
userValidFlag	int(2)	权限标志位
userType	int(2)	用户类型

表 2 仪器表核心表结构

字段名	字段类型	字段释义
apparatusID	int(11)	仪器 ID
apparatusName	varchar(32)	仪器名
belongAreaCenterID	int(11)	所属区域中心 ID
belongInstituteID	int(11)	所属研究院 ID
apparatusState	int(2)	仪器状态标志位
applyType	int(2)	仪器支持的预约模式

表 3 预约表核心表结构

字段名	字段类型	字段释义
applyID	int(11)	预约记录 ID
apparatusID	int(11)	被预约仪器 ID
apparatusApplyNo	varchar(32)	预约单号
bookManID	int(11)	预约用户 ID
startDate	datetime	预约单开始时间
endDate	datetime	预约单结束时间

根据上面对用户表、仪器表和预约表的核心表字段结构的分析, 根据聚类划分标准的不同, 可以分为按照所属研究所划分或按照所属区域中心划分. 但是无论采用哪种方法进行聚类分析, 特征变量均包含以下两个方面.

(1) 能够标识用户, 例如用户所属研究院 ID 和用户所属区域中心 ID.

(2) 能够标识仪器, 例如仪器所属研究院 ID 和仪器所属区域中心 ID.

同时, 在聚类时不仅需要选取合理的特征变量, 而且还需要在数据清洗和数据预处理阶段保证数据的正确性. 例如, 需要保证用户具有预约权限、仪器能够被预约、预约的时间段有效等.

2.1 聚类分析分表

根据选取的特征变量可以将预约表中的每一条数据使用一个二维向量表示. 根据 SAMP 系统实际业务架构可知, 预约用户与被预约的仪器均以区域中心为基准呈现一定的聚集性, 因此本文采用 K-means 聚类算法按照区域中心对预约数据进行聚类.

K-means 聚类算法通过迭代计算各个点与质点之间欧氏距离的平均值, 将数据集 D 中的数据划分到 K 个簇 $C_1, \cdots, C_k$ 中, 使得对于 $1 \le i, j \le k$, $C_i \subset D$ 且 $C_i \cap C_j = \Phi$. 这样的划分结果使得每一个簇内的对象相互相似, 而簇与簇之间对象存在差异得到最终分类结果. K-means 聚类算法是一种基于形心的划分算法, 在得到最终 K 个类的同时也得到 K 个形心 C_i, 数据对象 $P \in C_i$ 与该簇的代表 C_i 之差用 $dist(P, C_i)$ 度量, 其中 $dist(x, y)$ 是两个点 x 和 y 之间的欧式距离, 因此聚类目标是使得各类的聚类平方和最小, 即最小化公式如下:

$$E = Min\left(\sum_{i=1}^{k}\sum_{p\in C_i} dist(P,C_i)^2\right) \tag{1}$$

K-means 聚类算法的时间复杂度与预约表中的数据总量 N, 聚类后的簇数 K, 以及程序的迭代次数 t 具有线性关系. 结合实际应用场景中的数据规模可知, K 和 t 为常数级别且均远小于 N. 因此当预约数据总量 N 逐渐变大时, K-means 聚类算法相对可伸缩且有效[13].

根据聚类结果将预约表数据依次划分到各个分表中. 为保证预约表数据读写请求能够准确定位到各个分表, 需要在划分的过程中动态构建分类映射和分类路由. 分类映射由分类结果表和分表结果表构成. 其中分类结果表的数据在聚类完成后填充的, 记录了聚类结果的类标号、核心点坐标以及聚集类的规模. 分表结果表是在聚类的过程中动态填充的, 主要记录了子表标号、子表表名、分类标号以及子表规模. 而分类结果表中的类标号与分表结果表中的类标号具有主外键关系. 分类结果表和分表结果表的表结构如表 4、表 5 所示.

由于分表的策略是根据区域中心 ID 进行聚类划分, 因此无论从何种实体角度对预约表进行查询最终都会将查询对象按照其所属的区域中心转换成去相应分表中查询的方式. 因此分表路由表保存了每一个区域中心被哪些簇所包含, 分表路由表结构如表 6 所示.

表 4 分类结果表表结构

字段名	字段类型	字段释义
classificationID	int(11)	聚类结果的类标号
coordinate	varchar(32)	核心点坐标
classificationScale	int(11)	聚类的规模

表 5 分表结果表表结构

字段名	字段类型	字段释义
subtabulationID	int(11)	子表标号
subtabulationName	varchar(32)	子表表名
classificationID	int(11)	聚类结果的类标号
subtabulationScale	int(11)	子表的数据容量规模
subtabulationAvailable	int(2)	是否在子表中添加数据:

表 6 分表路由表表结构

字段名	字段类型	字段释义
routeID	int(11)	分表路由表 ID
belongAreaCenterID	int(11)	所属区域中心 ID
classificationID	int(11)	聚类结果的类标号

当预约表中插入新数据时, 要对新插入的数据利用聚类模型进行聚类, 步骤如下:

(1) 获取新数据的聚类二维特征点 $P(x, y)$.

(2) 计算 P 与聚类模型中各个形心 $C_1, \cdots, C_k$ 的欧式距离 $d_1, \cdots, d_k$, 将 P 划分到与各个形心欧式距离最小的簇 C_p 中, 并更新 C_p 的形心坐标.

(3) 更新分类映射和分表路由数据. 如果该条预约记录中涉及到的区域中心 ID 与簇 C_p 的映射在分表路由表中不存在, 则将映射关系添加到分表路由表中.

(4) 更新分类结果中簇 C_p 的形心坐标 (x_p, y_p) 和该簇的聚类规模 M_p 如下:

$$\begin{cases} x_p = \dfrac{x + x_p \times M_p}{M_p + 1} \\ y_p = \dfrac{y + y_p \times M_p}{M_p + 1} \\ M_p = M_p + 1 \end{cases} \tag{2}$$

2.2 分表后的扩容策略

即使将预约表进行分表处理后, 每一张子表的数据量也会随着时间增加而增加, 当子表的数据量增加到一定的数量级后, 子表的查询性能仍然会下降. 因此在分表的同时也需要为子表制定合理的扩容策略来解决由于子表数据量的增加而导致的查询性能下降问题.

评判分表扩容策略的标准是扩容过程中的数据迁移量. 在分表的过程中, 维护了分类映射和分表路由, 并且设置子表的负载因子 f, 若每一张子表的数据规模为 M, 则子表的阈值 S 与负载因子 f 和数据规模 M 具有如下关系:

$$S = M \times f \tag{3}$$

假设当前存在预约子表 T, T 所属的类标号为 C, 当 T 的数据容量达到阈值 N 后需要对 T 进行扩容处理. 系统的扩容策略为: 新建预约子表 T', 将其添加到子表结果表中, 设置 T'—所属的类标号为 C, 重置 T'的数据容量为零, 并将 T 在分表结果表中的数据添加标志位置零, 表示不再向 T 中添加数据. 这样当再有类标号为 C 的预约数据插入时, 会直接分配到 T'中, 而 T 中的原始数据仍然保存在 T 中, 从而达到了原始数据零迁移的分表扩容目的.

而当出现新的分类标号导致分表需要增加时, 首先将分类结果添加到分类结果表中, 创建与之映射的子表, 再将子表记录添加到分表结果表, 并更新分表路由, 最后将涉及到的区域中心 ID 与分类结果 ID 添加到分表路由表中, 同样也能实现零数据迁移扩容.

3 实验分析

本实验使用 SAMP 系统开发环境中仿真镜像数据库中存储的预约记录、用户信息及仪器信息. 服务器系统为 CentOS7, 数据库使用 MySQL 5.5，默认存储引擎 InnoDB.

3.1 实验数据说明与处理

数据分为两部分使用, 部分数据用于构建最小训练样本数据集, 对预约记录进行聚类分析; 剩余数据用于测试基于区域中心的聚类分表策略对系统查询性能的提升效果. 经过实验测试发现当预约记录的数据达到四万条即可对预约数据集进行有效划分, 因此用于构建最小训练样本数据集的数据量 N 的值为:

$$N = 40000 \times K;\ K\text{为簇数} \tag{4}$$

实验对用户表、仪器表和预约表进行聚类分析时, 重点采集了包括用户 ID、用户所属区域中心 ID、用户权限标志位、仪器 ID、仪器所属区域中心 ID、仪器状态标志位、预约单开始时间、预约单结束时间和预约记录 ID 在内的九维特征属性共同对训练样本数据集进行处理和分析, 并分析得出这些特征属性具有以下特点:

(1) 区域中心 ID 为一个取值在[10, 24]内的整数, 其中处于[10, 23]区间内的整数表示现有的 14 个区域

中心 ID, 区域中心 ID 值为 24 表示院外单位.

(2) 用户 ID 号是一个八位整数, 其中前两位是用户所属的区域中心 ID, 中间两位是用户所属的研究院 ID, 后四位是递增的数字.

(3) 仪器 ID 同样是一个八位整数, 不同于用户 ID, 仪器 ID 的前两位是仪器所属区域中心 ID 乘以四, 后六位表示的含义与用户 ID 号相同.

在聚类前需要对样本数据进行数据清洗与预处理确保数据的正确性, 具体过程如下:

(1) 去除用户权限标志位为 0 的预约记录, 即去除无预约权限用户的预约记录.

(2) 去除仪器状态标志位为 0 的预约记录, 即去除预约无效仪器的预约记录.

(3) 根据每一条预约记录的开始时间和结束时间, 去除预约时间少于 0.5 小时的预约记录.

最终每一条预约记录由一个三维向量 (预约记录 ID, 用户 ID, 仪器 ID) 唯一表示, 并使用其中的后两维属性进行聚类.

3.2 实验方法设计

本实验对训练样本集使用 K-means 算法根据用户所属区域中心 ID 和仪器所属区域中心 ID 进行聚类分析并根据聚类结果将数据库中的预约表进行水平切分. 实验构建的训练样本集数据量为 669010. 同时根据每一条预约记录的预约 ID 在聚类的过程中动态构建分类结果表和分表结果表, 并在完成聚类后构建分表路由表. 其初始聚类结果如图 2 所示.

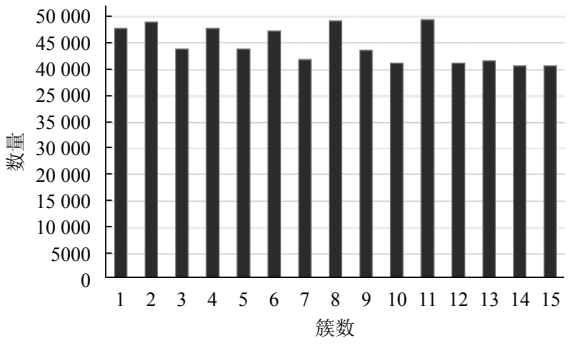

图 2 初始聚类结果图

完成聚类模型的构建后, 配合分类映射与分类路由以及分表扩容策略对全部预约数据表进行分表. 为了测试分表对整体数据表查询性能的提升, 实验定义三组查询测试语句分别从用户、仪器以及并发预约查询三方面进行测试. 设计实验测试如下.

分表前, 查询事件在预约总表中查询, 获取查询结果, 记录查询时间. 分表后, 查询事件通过以下几步获取查询结果, 记录查询时间:

(1) 获取查询对象所属区域中心 ID.

(2) 在分类路由中查询包含此区域中心 ID 的子表 ID.

(3) 在各子表中进行查询.

在同等实验条件前提下, 通过对三组查询测试语句执行时间的比较来分析聚类分表对查询性能的提升效果, 设分表前查询平均执行时间为 t_1, 分表后查询平均执行时间为 t_2, 则分表查询提升效率 f 为:

$$f = \frac{\overline{t_1} - \overline{t_2}}{\overline{t_1}} \tag{5}$$

本实验对查询时间及查询性能的测试均忽悠数据库对相同查询结果的缓存影响, 即在 MySQL 数据库中对查询语句添加“sql_no_cache”关键字来禁用查询缓存, 从而多次实验取平均执行时间.

3.3 实验结果分析

实验从用户角度进行查询测试, 选取查询事件: 查询用户 ID 为 userID 的用户预约的所有仪器. 分表前实验设置在禁用查询缓存的条件下执行该查询 100 次, 平均查询时间 0.364 秒, 具体查询时间分布如图 3 所示.

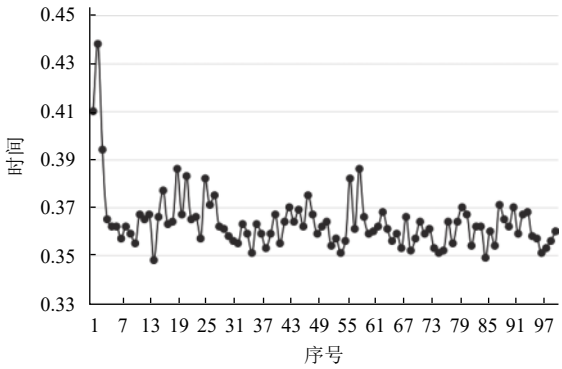

图 3 分表前用户预约记录查询时间分布图

将预约表进行分表处理后, 在同等实验条件下设置执行该查询 100 次, 平均查询时间 0.111 秒, 具体查询时间分布如图 4 所示.

实验从仪器角度进行查询测试, 选取查询事件: 查询仪器 ID 为 apparatusID 的仪器所有被预约记录. 分表前实验设置在禁用查询缓存的条件下执行该查询

100 次, 平均查询时间 0.355 秒, 具体查询时间分布如图 5 所示.

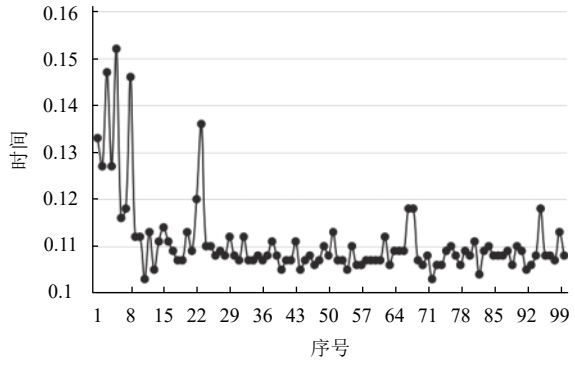

图 4 分表后用户预约记录查询时间分布图

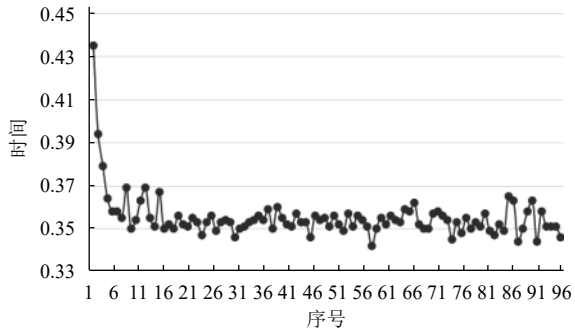

图 5 分表前仪器被预约记录查询时间分布图

将预约表进行分表处理后, 在同等实验条件下设置执行该查询 100 次, 平均查询时间 0.075 秒, 具体查询时间分布如图 6 所示.

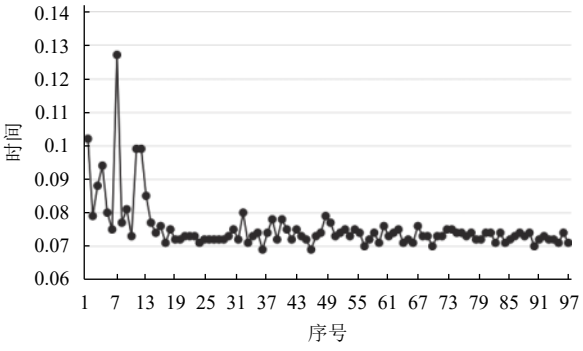

图 6 分表后仪器被预约记录查询时间分布图

实验选取并发查询事件为: 并发查询仪器 ID 为 apparatusID 的仪器被预约的记录. 实验使用 Apache JMeter 工具模拟多用户同时预约的并发场景, 设置所有线程数在一秒内全部启动, 设置数据库连接池的最大连接数为 50, 最大等待连接时间为 120 秒, 并配置数据库驱动与 JDBC 连接参数. 根据实际 SAMP 系统运行经验, 设置线程并发数为 1000, 即设置了在同一时间段中的最大并发量为 1000. 则分表前后对并发查询事件的查询性能统计如表 7 所示.

表 7 分表前后对并发查询事件的查询性能统计

	Average(s)	Median(s)	Min(s)	Max(s)	Throughput(/s)
分表前	58.9	58.3	0.8	120.3	3.5
分表后	6.2	6.5	0.2	11.4	79.1

表 7 中各个字段的含义如下:

(1) Average: 查询的平均响应时间.

(2) Median: 响应时间的中间值, 即在全部请求中一半请求的响应时间低于该值, 一半请求的响应时间高于该值.

(3) Min: 请求的最短响应时间.

(4) Max: 请求的最长响应时间.

(5) Throughput: 吞吐量, 即每秒处理的请求数.

由上述实验可知, 根据 SAMP 系统实际区域中心 ID 进行聚类分析并按照聚类结果对预约表进行分表处理后, 从预约用户角度对预约表进行查询操作的查询性能提升率为 69.5%; 从被预约的仪器设备角度对预约表进行查询操作的查询性能提升率为 78.9%; 从并发查询角度对预约表进行查询操作的查询性能提升率为 88.9%.

4 结论与展望

在 SAMP 系统开发环境仿真数据库中应用基于 K-means 聚类的数据库分表策略, 其中包含的的聚类策略以及分表扩容策略均在 SAMP 系统的应用中具有实际意义, 因此弥补了当前主流数据库分表策略 (例如 Hash 取模分表策略) 在切分预约表时的不足, 对预约表数据的查询性能提升了 70%, 达到了优化数据库查询性能的目的, 解决了 SAMP 系统整体系统性能不高的问题. 由于现运行的 SAMP 系统每天所处理的并发请求以及对数据的读写操作较仿真系统而言规模更加庞大复杂，因此将该分表策略应用于现运行的 SAMP 系统时对系统性能的提升率要通过系统运行状态做进一步分析.

参考文献

1 韦美雁, 段华斌, 周新林. 大数据环境下的 MySQL 优化技术探讨. 现代计算机, 2018, (10): 68–72.

2 Herodotou H, Borisov N, Babu S. Query optimization techniques for partitioned tables. Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data. Athens, Greece. 2011. 49–60.

3 刘阳娜. 大数据下的 MySQL 数据库的效率优化. 信息通信, 2017, (12): 111–112. [doi: 10.3969/j.issn.1673-1131.2017.12.049]

4 刘晓光. 基于 MySQL 的分布式 SQL 数据库的设计与实现[硕士学位论文]. 北京: 中国科学院大学, 2016.

5 韩兵, 李晶晶, 方英兰. 基于 JDBC 数据管理与查询优化的研究. 计算机技术与发展, 2018, 28(9): 176–180. [doi: 10.3969/j.issn.1673-629X.2018.09.036]

6 赵曦, 李颖, 徐江. 利用垂直分割技术的分布式数据库设计研究. 控制工程, 2018, 25(1): 154–159.

7 Khan M, Khan MNA. Exploring query optimization techniques in relational databases. International Journal of Database Theory and Application, 2013, 6(3): 11–20.

8 董献伦. 基于关系型数据库的数据切分问题研究[硕士学位论文]. 济南: 山东大学, 2016.

9 王照清. 大数据环境下数据查询优化技术应用研究[硕士学位论文]. 北京: 北方工业大学, 2016.

10 韩兵, 王照清, 廖联军. 基于 MySQL 多表分页查询优化技术. 计算机系统应用, 2016, 25(8): 171–175.

11 任满杰, 何文义, 付华. 数据库分割技术及其对数据库系统的影响. 阜新矿业学院学报 (自然科学版), 1994, 13(4): 102–105.

12 孙伟东, 夏秀峰, 马宗民. 利用数据库实现分布式任务的程序和数据存储. 航空电子技术, 2009, 40(1): 16–19. [doi: 10.3969/j.issn.1006-141X.2009.01.004]

13 梁双, 周丽华, 杨培忠. 基于聚类分析分库策略的社交网络数据库查询性能与数据迁移. 计算机应用, 2017, 37(3): 673–679.

14 孙辉. MySQL 查询优化的研究和改进[硕士学位论文]. 武汉: 华中科技大学, 2017.

15 吴金朋. 一种大数据存储模型的研究与应用[硕士学位论文]. 北京: 北京邮电大学, 2013.