DOI:10.15966/j.cnki.dnydx.2022.06.007

基于MySQL的数据库查询性能优化技术研究

王 景

（甘肃交通职业技术学院，甘肃 兰州 730070）

摘 要：面对小范围局域网内数据总量不断增长的发展趋势，依托MySQL数据库管理系统、SQL查询语句，进行数据表索引、SQL查询语句、分页查询方式、MySQL查询缓存等设置，随后开展后台数据库中数据信息的SQL语法规则分析、扫描、预处理、查询执行、存储操作，针对影响数据库查询效率的客观因素，提出数据访问响应、页面加载的性能优化方案，保证MySQL数据库查询及数据调取的性能。

关键词：MySQL数据库；数据查询；性能优化；方法

中图分类号：TP392 **文献标识码：**A **文章编号：**1008 - 6609 (2022) 06 - 0090 - 04

1 引言

当下局域网内的数据海量化、数据库查询效率极低，且以SQL语句为主的数据查询指令执行过程中，存在着数据表索引创建滞后、数据遍历方式不合理、数据类型与检索条件不匹配、数据表联接顺序不准确的问题。为达成数据信息查询过程中后台数据库、磁盘I/O性能调取与优化的目标，本文提出依托MySQL、HBase数据库、网络微控制器、磁盘I/O等软硬件，以MySQL关系型数据库、数据表作为查询语言，进行企业内部业务数据查询的方案，将网络业务数据的处理增加至50、100及以上的执行线程，保证在短时间内完成数据表索引、SQL查询语句、分页查询方式、MySQL查询缓存的操作执行，并合理利用MySQL查询缓存的处理方式，降低MySQL数据库对后台CPU、磁盘资源的消耗。

面对小范围局域网内数据总量不断增长的发展趋势，徐昂[1]提出优化SQL检索索引的方式，用于提升后台数据库的信息查询效率；乐艺[2]提出改进的布谷鸟搜索算法，设置数据查询优化的多个约束条件，大幅度改善数据库查询的效率和性能。岳彬森[3]、陈年飞[4]等针对MySQL数据库中的数据索引查询，提出BTree数据结构、系统索引结构、磁盘存储的优化策略。

本文通过以MySQL关系型数据库、数据表形式，作为结构化数据信息查询的语言，设置SQL查询语句、有效索引、字符集及用户、分页查询模式、查询缓存等功能，并在微控制器、磁盘I/O等的支持下进行联表查询，优化数据信息查询的性能，尽可能减少信息查询过程中的数据库、磁盘I/O性能占用，提升外部用户数据访问、检索的响应速率和准确率。

2 MySQL数据库信息查询的测试环境搭建

MySQL数据库对后台服务器的CPU、内存、I/O等要求较高，因而本文选取IBMX3850系列处理器、IBM DS3512磁盘阵列，每台磁盘阵列中配置8块2T SAS硬盘，作为数据库服务器，测试软件使用“Windows+MySQL5.5”的关系型数据库管理系统。在此基础上安装MySQL软件包，以及MySQL-debuginfo、MySQL-devel、MySQL-server共享库等的组件[5]，具体数据库信息查询的MySQL测试环境搭建流程如下：

（1）新建用户并以安全方式运行安装进程，执行代码为：groupadd -r mysql；useradd -g mysql -r -s /sbin/nologin -M -d /mydata/data mysql。

（2）将MySQL5.5.28版本安装至本地，并对MySQL软件作初始化设置。

（3）为MySQL软件提供主配置文件：cd /usr/local/mysql；cp support-files/my-large.cnf /etc/my.cnf。添加MySQL数据文件的存放位置：datadir = /mydata/data。

（4）为MySQL软件提供sysv服务脚本：cd /usr/local/mysql；cp support-files/mysql.server /etc/rc.d/init.d/mysqld。

（5）添加相应脚本至服务列表，启动服务测试：chkconfig --add mysqld；chkconfig mysqld on。

3 基于MySQL关系型数据库的SQL语句信息查询执行流程

MySQL数据库通常以SQL查询语句，作为后台库内数据表中一系列数据分组、数据添加、字段排序的执行语句，分为SQL输入—词法扫描—语法分析—语义预处理—SQL优

作者简介：王景（1978—），女，甘肃兰州人，硕士，副教授，研究方向为计算机应用。

化执行的流程，具体如图1所示。

（1）根据外部用户的数据访问、检索查询业务的逻辑要求，编写用于信息查询的SQL语句。

（2）在SQL语法规则支持下，利用词法扫描器、语法分析器等设备，识别出SQL语句中包含的字符、字符串、单词、空格等操作符，判断SQL语句的关键字、关键词、引号等的顺序及匹配是否正确，若正确则生成语法分析树。[6]

（3）随后由预处理器为主导，进行树中各节点语法的检查，检验生成语法分析树的合法性，并生成新的语法分析树，并将数据库对象重名、别名、不存在等的错误信号，向用户客户端返回报告，但整体解析树的结构保持不变。

（4）利用查询优化器、以关系代数为基础，对新的语法分析树作出逻辑、物理优化，进行语法分析树中各节点的语法调整，以及信息查询顺序、扫描方式、联接算法等调整后，生成语法查询树。

（5）调用存储引擎API、依据语法查询树，执行网络数据的查询指令，并将最终的查询执行结果返回至用户客户端，即完成相应数据信息的查询操作。

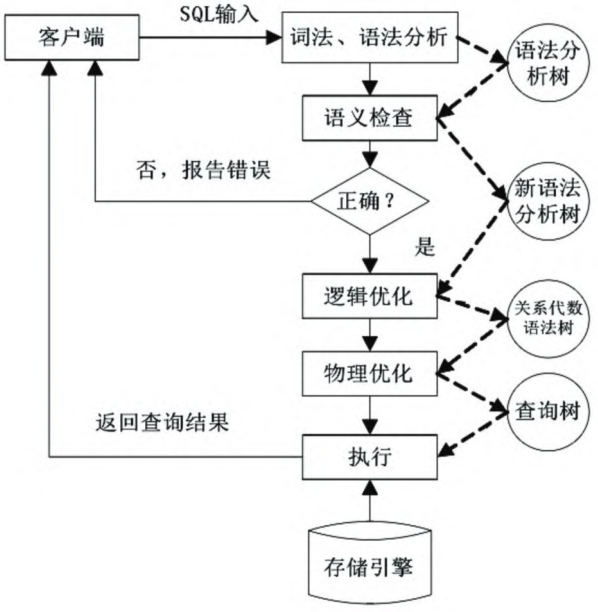

图1　MySQL数据库的SQL语句信息查询执行流程

4　MySQL数据库查询性能的影响因素分析

MySQL关系型数据库中，使用词法扫描器、语法分析器、预处理器、查询优化器等设备，以及SQL语句进行网络信息查询时，通过采取不同的执行策略，包括对CPU利用率、I/O通信等待时长、网络数据传输等调用方式，作出适宜的调整设置，可有效降低数据库信息的查询成本。当下有关MySQL数据库查询性能的影响因素，主要包含索引创建、数据遍历方式、数据类型兼容性、数据表联接顺序等内容。

4.1　数据表索引创建的滞后

从特定数据提取关键字、关键词，且定义不同数据之间的对应关系、映射关系，根据数据表索引关系进行关键字、关键词、符号等字段的算法排序，可辅助存储引擎API快速找到用户需要的数据内容，因而MySQL索引的建立，对后台数据库的信息查询速率、查询质量而言至关重要。

但部分MySQL数据库的Web网站空间信息查询，却未创建以SQL语句为主的适合索引，存储引擎API也不能利用数据表索引，进行where、having、order by等子句的关键列、关键字查询，而只能依托后台微处理器、存储器、I/O端口被迫执行全表扫描，增加阵列磁盘的荷载负担。[7]

4.2　数据遍历方式的不合理

当前在MySQL数据库中结构性数据的查询，通常会使用到for、for... in、for... of、while、do... while等的SQL语句，进行某一行、列数据的集中遍历，但在对关键字、关键词索引数据(index)遍历的过程中，可能会由于某些错误(value)变量的未声明，造成非必要数据的遍历、重复性遍历。[8]如以下for语句的数据查询遍历代码中，部分“List的待删除元素”实际未被遍历，而其他非必要数据被遍历，由此延长数据访问的响应时间、降低索引对字段的检索准确率。

```
private void remove(List<数据对象> tempData) {
    if (ArrayUtils.isNotEmpty(tempData)) {
        for (int i = 0; i < 待删除的list集合.size(); i++) {
            for (int j = 0; j < tempData.size(); j++) {
                if (待删除的list集合.get(i).getId().equals(tempData.get(j).getId())) {
                    tempData.remove(j);
                    totalNum--;
                }
            }
        }
    }
}
```

4.3　数据类型与检索条件的不匹配

MySQL库内的数据类型与检索条件不匹配，是数据表索引检索面临的另一问题，如使用“<>”“in”“not”“or”等的运算符，以及选择smallint、int等的数据类型，则极大可能会增加全表扫描操作符的概率，增大磁盘存储的占用空间。同时在检索条件中引入“<>”“in”“not”“or”等的操作符，将导致表内使用属性列、空值null进行索引判断，以至于降低数据字符类、字段的查询速率。而一旦非兼容数据类型、特定检索条件匹配完成后，则难以使用查询优化器，作出进一步的语

法逻辑分析、物理分析优化操作。

4.4 数据表联接顺序的不准确

数据表联接顺序对MySQL数据库的查询性能至关重要，但部分数据表的行列联接，并未考虑到items表、stock表等数据表中相应for、for... in、while子句长短的差异性。如通常情况下stock表更短、数据表联接顺序也更加合理，因而无需调用过多的后台微处理器、存储器、I/O接口负载，就可以完成局部指定数据的属性列函数计算、索引表扫描。若采取items表的数据联接顺序，则很大可能会造成数据查询中传递的总行数增加、数据表联接速率降低。

5 MySQL数据库查询性能的实验测试与结果

通过基于某一企业的内部业务数据，围绕MySQL、HBase、OpenTSDB等类型的数据库，开展MySQL数据库查询性能的实验测试，比较不同类别数据库的查询性能差异。在以上三类数据库总数据条数、线程数存在差异的情况下，要求其进行企业内部业务数据的1000次查询实验，5次实验查询的数据量分别为1000、10000、39000、50000和100000，启动的数据库线程条数分别为1、10、50和100条，得出的数据库查询性能结果如图2所示。[9]

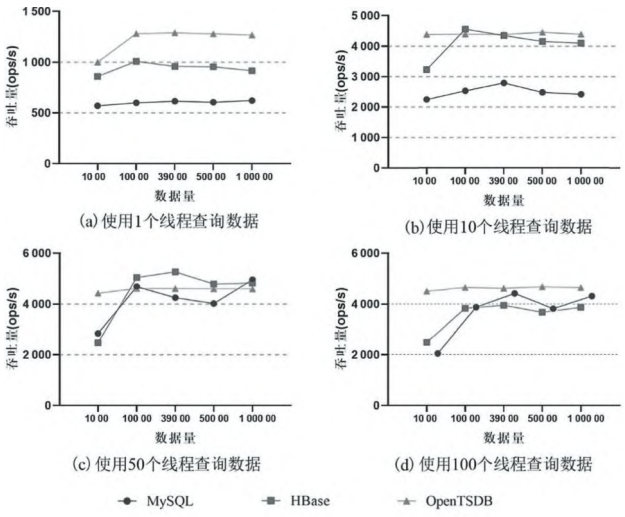

图2 三类数据库的查询性能的实验测试结果

从图2中不同类型数据库的查询性能测试结果可以得出：当所需要查询的内部业务数据量较小情况下，使用MySQL数据库查询操作的吞吐量处于较低水平，不如HBase、OpenTSDB数据库的数据查询吞吐量。但当网络业务数据处理的执行线程，增加至50、100个及以上的状况下，MySQL数据库的数据查询吞吐量将快速增长，且基本与HBase、OpenTSDB数据库的吞吐量持平，但MySQL数据库对后台CPU、磁盘资源的消耗更为严重，仍旧存在进一步的性能优化空间。

6 基于MySQL数据库的网络数据查询性能优化策略研究

6.1 创建有效数据表索引

MySQL数据库中数据表索引的创建，是采用语法分析法开展物理查询技术优化的重中之重。通过创建包含数据行列值、存储位置的索引表格，可辅助MySQL数据库应用程序，利用扫描索引数据表方式，包括主/外键属性行、属性列扫描，搜索外部用户想快速找到的数据内容，而无需对全表进行全盘扫描。[10]

如针对MySQL数据库内的c1、c2、c3共10万条数据，使用SELECT c1,c2,c3 FROM t1 WHERE c1(c2,c3)=50001的查询语句，查询测试数据表t1中的第50001条数据记录，得出未建立索引、建立主键索引后的查询用时(如图3所示)，可以看出在c1(c2,c3)查询条件中建立索引后，查询用时缩短至0.001s。

```
mysql> SELECT c1,c2,c3 FROM t1 WHERE c1=50001;
+-------+-------+-------+
| c1    | c2    | c3    |
+-------+-------+-------+
| 50001 | 50001 | 50001 |
+-------+-------+-------+
1 row in set (0.06 sec)
mysql> SELECT c1,c2,c3 FROM t1 WHERE c1=50001;
+-------+-------+-------+
| c1    | c2    | c3    |
+-------+-------+-------+
| 50001 | 50001 | 50001 |
+-------+-------+-------+
1 row in set (0.0001 sec)
```

图3 未建立索引、建立索引后的查询用时

6.2 SQL语句优化

SQL语句作为网络数据逻辑查询技术优化的手段，主要依据SQL语法查询规则、关系代数理论，对SQL语句作出等价转换。如between……and、IN、OR、LIKE等的关键运算符，不支持条件判断的索引扫描，因而可以用其他诸如>=、<、>=and <的运算符进行转换。或者将未进行数据分组、数据排序的SQL语句，重写为多数据表连接的等价语句，这样可以将子查询的数据过滤条件与父查询的过滤条件形成对接，减少某一个(类)数据查询的执行次数。

6.3 分页查询优化

当某一局域网内的业务数据信息较多时，仅仅采用全表查询检索方式，其本身的数据查询效率、查询质量非常低，且向用户客户端一次性展示过多数据，将造成查询页面的出错或卡死。[11]因此，利用多次查询显示的“分页查询”方式，包括选用数据ID限定、数据条数限定的方法，设置Pic表查询的id字段长度、数据条数长度。

如分别从某一数据表的100行、1000行、10000行处，查询前10行、200~300行的数据，可提高多次查询的响应速度、

缩短查询时间，具体在数据ID限定、条数限定状况下的分页查询性能如图4所示。因而可以看出，数据ID限定查询的响应时间控制效果，要明显优于数据条数限定的查询方案。

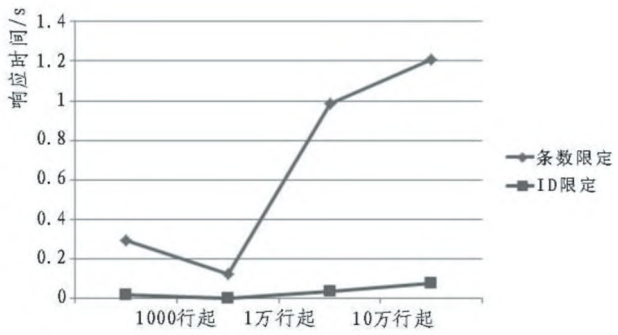

图4　MySQL数据库在数据ID限定、条数限定下的分页查询性能

6.4　合理利用MySQL查询缓存

Query Cache查询缓存重用的优化技术，是MySQL数据库中数据信息分析、保存的重要技术，其主要针对SQL查询语句的重复性提交情况，由MySQL数据库应用程序直接从查询缓存中检索并返回需要的信息查询结果，这样既能够减少某一个(类)数据的重复性查询，又将最大程度降低后台数据库的工作负载。

如通过使用SHOW VARIABLES LIKE '% QUERY_CACHE%'的数据查询语句，对MySQL库内的缓存数据进行查询，可完成数据表内大批量、相同的数据查询操作，但需注意在数据表结构、内容频繁被修改的情况，若使用MySQL缓存查询，则可能导致查询结果的失效。

7　结语

MySQL是当下应用最广泛的关系型开源数据库，其主要具有源代码可移植性、磁盘空间占用少、运行速度快等优势，可被用于中小型网站的Web后台数据信息查询、增删、修改操作。因此，不同企事业单位网络数据的查询，要在后台数据库硬件设备支持下，合理设置优化SQL查询语句、数据索引创建、遍历方式、数据类型兼容与匹配等要素，提升MySQL数据库的网络查询性能。

参考文献：

[1] 徐昂，成科扬. 基于关系型数据库的SQL检索优化研究[J]. 电子设计工程，2019(11)：51-55.

[2] 乐艺. 大规模数据库查询优化算法的设计与研究[J]. 科技通报，2019(09)：66-69+74.

[3] 岳彬森. SQL索引及SQL语句的应用技巧分析[J]. 现代信息科技，2019(19)：26-27.

[4] 陈年飞，王麒森，王志勃. MySQL数据库中关于索引的研究[J]. 信息与电脑(理论版)，2019(05)：175-176.

[5] 汤亚宸，方定江，韩海韵，等. 基于图数据库和知识图谱的电力设备质量综合管理系统研究[J]. 供用电，2019(11)：35-40.

[6] 韦美雁，段华斌，周新林. 大数据环境下的MySQL优化技术探讨[J]. 现代计算机(专业版)，2018(30)：68-72.

[7] 罗希意，霍晓阳，傅洛伊. 基于窗口函数和分布式集群的可视化学术搜索系统数据查询优化[J]. 上海交通大学学报，2019(08)：978-982.

[8] 马跃，王喆峰，尹震宇，等. 基于K-means的SAMP系统数据库查询性能优化策略[J]. 计算机系统应用，2019(06)：69-75.

[9] 和士琦，刘伟. 数字化信息平台Oracle数据库空闲碎片整合研究[J]. 自动化与仪器仪表，2020(08)：158-161.

[10] 李艳杰. MySQL数据库下存储过程的设计与应用[J]. 信息技术与信息化，2021(01)：96-97.

[11] 石丽怡. MySQL数据库字符集的问题研究[J]. 电子技术与软件工程，2020(12)：149-150.

Research on the Database Query Performance Optimization Technology Based on MySQL

WANG Jing

(Gansu Transportation Vocational and Technical College, Lanzhou 730070, Gansu)

Abstract: In the face of the growing development trend of the total amount of data in a small-scale LAN, relying on MySQL database management system and SQL query statements, this paper sets the data table index, SQL query statements, paging query mode and MySQL query cache, and then carries out SQL syntax rule analysis, scanning, preprocessing, query execution and storage of data information in the background database, aiming at the objective factors affecting the efficiency of database query. It puts forward the performance optimization scheme of data access response and page loading to ensure the performance of MySQL database query and data retrieval.

Keywords: MySQL database; data query; performance optimization; method