基于 B*树聚簇索引的加密字符串查询方法①

刘 洁

(江苏科技大学 计算机科学与工程学院, 镇江 212003)

摘 要: 为了提高在数据库中查询加密字符串数据的性能, 提出一种在索引特征值上创建 B*树聚簇索引的查询方法. 每一个待加密字符串数据对应一个索引特征值, 索引特征值以数值的形式保存在索引字段中. 查询时使用两阶段查询策略, 首先利用索引字段对加密数据进行一次粗糙查询过滤掉不相干的记录, 然后在返回的粗糙集合解密的基础上进行明文查询, 得到最终结果. 实验表明该方法较现有查询方法在查询性能有较大的提升.

关键词: 数据库加密; 加密字符串查询; 索引特征值; B*树聚簇索引

Practical Techniques for Querying over Encrypted Character String Based on B* Tree Cluster Index

LIU Jie

(School of Computer Science and Engineering, Jiangsu University of Science and Technology, Zhenjiang 212003, China)

Abstract: To improve the performance of querying over encrypted character string in database, this paper proposes a method that creating B* tree cluster index base on index characteristic code. Every encrypted data has an index characteristic code which store in an index filed as index. When querying the encrypted character data, applies the principle of two_phase query. In the first place, make use of index characteristic code to filter the records which is not related to the querying condition. Secondly, decrypt the rest records and take advantage of plaintext querying condition to get the final records. Results of experiments validate the performance of our method compared with exsiting ways at present.

Key words: database encryption; encrypted character string query; index characteristic code; B* tree cluster index

随着技术的飞速发展, 信息系统日益复杂化, 人们对计算机安全的需求与日俱增. 数据库作为当前网络和信息系统的基础平台, 解决数据库安全问题成为目前最为紧迫的挑战之一. 解决这问题的最有效的办法是对数据库中的重要数据进行加密处理. 现有的数据库系统已采取相关的安全措施, 但是仍然存在一些系统安全漏洞, 如黑客攻击[1]、DBA 的权限过大、备份介质丢失造成机密信息的泄漏等. 因此有必要对数据库中存储的机密数据进行加密处理[1,2]. 这样, 攻击者即使绕过了各种系统安全机制, 最后得到的也是密文数据, 同样也防止了备份数据流失造成的重要信息的泄密.

对数据库字符串数据进行加密是保证了数据库数据的安全, 但是使数据失去了本身固有的有序性、相似性、可比性的特性发生了变化[3]. 这样导致对字符串的查询变成了一大难题. 在一般现有的数据库中对加密字符串数据进行查询时, 需要先把所有的加密的字符串数据解密, 然后在得到的明文的基础上进行查询. 这样加/解密操作开销巨大, 极大地降低了系统的查询性能[4]. 另外, 对字符串数据加密后也加大了字符串模糊查询的难度.

针对以上问题, 王正飞[6]等人的研究工作给出了比较好的解决方案, 后来的崔宾阁[7]等人在王正飞方法上进行改进, 其查询效率基本与文献[6]持平. 曹杨等人[8]在王正飞等人的研究工作的基础上提出了一种基于对偶特征码的快速查询方法, 其伪记录过滤性能高于王正飞的方法, 查询效率也有提升. 本文提出了一种在索引特征值上构建 B*树聚簇索引的查询方法,

① 收稿时间:2012-11-04;收到修改稿时间:2012-12-17

相比文[8]查询效率有了进一步提高. 以下将文献[8]中的方法称曹杨方法.

1 查询结构

我们在应用程序和商用数据库中间增加了转换模块和过滤模块. 转换模块为上下层提供数据的转换功能, 将对明文的查询条件转换为对索引字段的查询条件. 过滤模块将返回的粗糙查询结果集合中过滤掉不满足查询条件的记录.

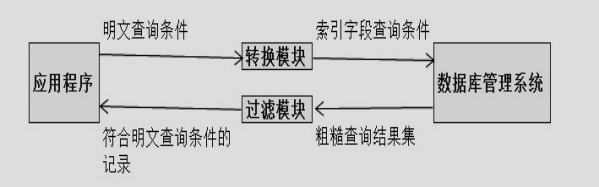

图 1　查询结构

2 基本概念

2.1 B*树聚簇索引

聚簇是指如果一组表有一些共同的列, 则这样一组表存储在相同的数据块中. B*树聚簇索引是传统 B*树索引的一个变体, 用于对聚簇键建立索引. 在传统的 B*树, 键都指向一行; 而 B*聚簇不同, 每一个聚簇键指向一个块, 其中包含与这个聚簇键相关的多行[9].

例如: 在图 2 中, 左边使用了传统的表, EMP 会存储在自己的段中. Dept 也存储在自己的段中. 他们可能位于不同的文件. 现在将值 10 抽取出来, 只存储一次. 这样聚簇的所有表中对应部门 10 的所有数据存储在这个块上. 这样当部门 10 进行查询时, 如果这些数据分布的到处都是, 就要花一些时间才能把他们收集起来. 如果这些数据都在一个块上, 通常就很容易得到. 聚簇有助于完成总是把数据联结在一起活着访问相关数据集(例如 10 部门中的每一个人)的读密集型操作. 聚簇表可以减少 oracle 必须缓存的块数.

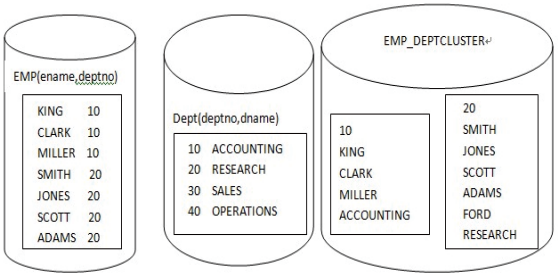

图 2　聚簇数据

2.2 索引特征值

索引特征值针对原明文字符串而言, 每一个明文都有一个与之相对应的数值型索引特征值. 索引特征值可以通过索引特征函数得到.

定义 1. 设有索引特征函数(IPCF): S1—>S2,其中 S1 为字符串 C1C2…Cn; S2 是无符号整数 b0b1…bm-1, 且初始值所有位为 0, n>m. H 为 hash 函数, 把 S1 中的两个相邻字符, 散列为 0~m-1 之间的一个数, 当且仅当某个 j, H(CjCj*1)=i 时, bi=bi+1, bi 直到为 9, 不再增加. 举例说明:

S1 为字符串 abcehklst,散列为 0~16 之间个数, 则 S2=IPCF(abcehklst)=(2010002010100100), 其中, 我们可以看出第一位和第七位上有两个重叠.

3 加密存储模式

对于传统关系模式 R(X1,X2,…,Xr,…,Xn), XR 为待加密的字符串字段, 加密后的关系模式为 R(X1,X2,…, XrE,…,Xn,Xrs)其中, XrE 是对应于 Xr 加密字符串字段, 即 XrE=E(Xr); Xrs 是对应于 Xr 的辅助索引字段, 用于存储索引特征值.

4 查询策略

本文使用两阶段查询[6]. 第一阶段: 粗糙查询, 利用索引字段对加密字段进行一次查询从而过滤掉与查询无相关的记录; 第二阶段: 精确查询, 对第一阶段得到的粗糙结果进行二次过滤得到符合明文查询条件的记录. 查询时, 将用户的明文查询条件, 通过转换模块, 转换成对相应的索引特征值的查询语句. 数据库系统利用索引特征码扫描索引, 得到符合条件的记录集. 再将得到的记录集通过过滤模块进行二次过滤, 得到符合用户明文条件的最终结果, 并以明文的形式返回给用户.

5 查询算法

5.1 完全匹配查询

查询与查询条件中的字符串值完全相等的记录. v_string 为查询条件中的字符串值, Xr 为加密字段, Xrs 为相应的索引字段, IPCF(v_string)=P1P2…PL, 则 where Xr=v_string 转换成 where $X_r^s = P_1 P_2 \cdots P_L$.

5.2 模糊匹配查询

查询包含查询条件中的字符串的记录. v_substring

为查询字符串值, Xr 为加密字段, Xrs 为相应的索引字段, ICPF($v_{substring}$)=P_1' P_2' …P_L', 转换 where $X_r^s > P_1' P_2' \cdots P_L'$ and 每位的差值>=0.

5.3 查询过滤算法

设经过索引查询得到结果记录集合为 Rset 包含 n 条记录, 分别记为 R1, R2, …, Rn. 先将它们解密为记录集合 RsetD. 在 RsetD 中执行对明文查询条件可以得到查询结果记录集, 即 where Xr like v_substring.

6 算法分析

6.1 安全性分析

在加密模式中, 待加密字段通过加入盐值分组加密算法加密后, 以密文的形式存储在数据库中, 即使有两个数据行包含相同的数据, 也不会这两行数据加密成相同的值. 这样可以避免探测型攻击, 从而保证它的安全性.

6.2 过滤效率分析

由于 IPCF 函数产生索引特征值拥有计数累加特性, 使得转换后的模糊匹配的查询条件更为严格. 但是, 索引特征函数(IPCF)的累加能力最大只能在 9. 对于不同长度的字符串, 本文方法和曹杨方法之间的过滤效率差异不一样. 如果面对索引特征值中多位数值上累计到达 9 或 9 以上的字符串, 相比曹杨方法, 还有一定的不足.

6.3 存储空间分析

相比传统模式, 加密模式中存储加密字符串, 另外还增加了索引字段, 所以增加存储空间的开销. 当索引字段位数比较大, 新增存储空间较大; 反之, 索引字段位数小, 新增存储空间较小[5,6]. 因为在索引字段上增加 B*树索引表, 与曹杨方法相比, 新方法消耗更多的存储空间.

7 实验与结果分析

实验主要目的是对比两种方法的查询性能. 根据 TPC-H 标准[10], 由 dbgen 标准程序自动生成实验数据库, 比例因子为 0.1. 以其中的 lineitem 表作为本次实验的数据源, 以 TCOMMENT 字段作为待加密的敏感字段. 加密算法采用 AES128. 实验环境为 Windows7(Intel core i5 2.5CHz+2GRAM)+ORACLE11G+MyEclipse8.5M. 由于数据库中数值型数据最长为 38 位, 所以在实验过程中索引特征值最长的位数为 38 位.

实验 1: 过滤性效率测试

定义 2. 如果数据表中的记录数为 N, 经过第一阶段粗糙查询返回的记录为 n1, 在第二阶段精确查询返回的记录为 n2,则过滤效应= $\frac{N-n1}{N-n2}$. 其中, N-n1 表示经过第一阶段过滤掉的记录. N-n2 表示不符合明文查询条件的记录[7].

实验 1 中测试查询策略中第一阶段过滤的效率. 过滤效应与特征值位数密切相关. 图 3 给出了对于完全匹配查询, 特征值位数变化对于过滤效率的影响. 随着特征值位数的增加, 本文方法和曹杨方法过滤效率都相应的增加, 并且他们的曲线是完全相同的. 当特征值位数增加到 16 为后, 过滤效率增加幅度减缓. 由于查询的过滤效率已经接近 100%, 出于存储空间考虑, 选择特征值位数为 16.

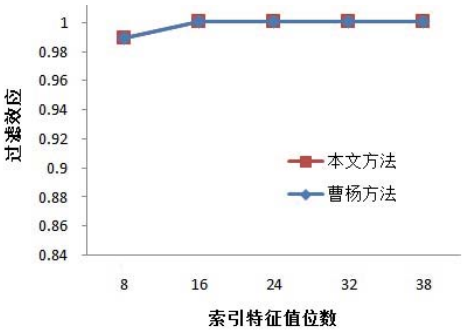

图 3 过滤效率测试

实验 2: 查询时间测试

实验 2 中, 测试本文方法和曹杨方法查询结果所花费的时间比较. 图 4 中给出了随着索引特征值位数不同, 两种查询方法所花费的时间. 图 4 为完全匹配查询条件的情况. 在图 4 中, 随着特征值位数的增加, 本文方法和曹杨方法所花费的时间越来越少.

由图 3 可知他们有相同的需要解密的记录数, 由于在本文方法中通过 B*树聚簇索引查询满足索引特征值条件返回记录, 不要进行全表扫描, 从而提高了查询的性能. 从图 4 中可以看出, 本文方法所需要花费的时间是曹杨方法的 1/6, 可见很大程度上提高了系统的性能.

8 结语

本文方法通过对加密数据表中增加索引字段, 在其上建立 B*树聚簇索引, 将对加密数据的查询转换为

对索引字段的查询, 在查询性能上较曹杨方法有较大的提高. 由于在数据库中数值型数据是有限制的, 最长是 38 位, 则本文方法只适合某一定长度以内的字符串.

本文方法还存在其他不足, 如模糊匹配查询(条件为 like)时, 本文方法和曹杨方法在过滤效率上仍然一致. 但是本文查询时间代价比曹杨方法多出一倍. 此时, 数据库系统采用全表扫描的方式查找需要解密查询的记录, B*树聚簇索引没有启用. 如何在模糊查询的时候仍然使用索引来提高查询性能, 需要进一步研究.

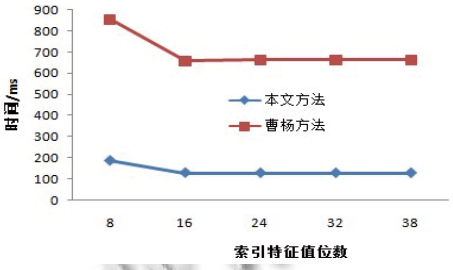

图 4 查询时间测试

参考文献

1 Canm M, Kantarcioglu M. Design and analysis of querying encrypted data in relationl database.IFIP WG11. 3Working Conference on Database and Applications Security. Berlin Springer, 2007.

2 Stalling W. Cryptography and network Security princeples andpractices. PrenticeHall, 2003: 14–71.

3 Jakodias. Database security and privacy. ACM Computer Surveys, 1996,28(1):129–131.

4 Stinson DR. Cryptography theory and practice. CRC Perss, 2002: 23–56.

5 王正飞,施伯乐.数据库加密技术及其应用研究[学位论文].上海:复旦大学,2005.

6 Wang ZF, Dai J, Wang W, et al. Fast query over encrypted character data in database. Communications in Information and System, 2004,3314(4):289–300.

7 崔宾阁,刘大昕,王桐.支持快速查询的数据库加密方法研究.计算机科学,2006,33(6):115–118.

8 曹杨,何大可.数据库加密字符串快速查询方法.计算机应用研究,2009,26(2):736–738.

9 Thomas Kyte.oracle 深入数据库体系结构.北京:清华大学出版社,2011.

10 Transaction Processing Performance Council TPC Benchmark TM H Standard Specification Revision. http://www.tpc.org.

(上接第 43 页)

核心指导思想[1]. 随着协同办公模式的不断推进及广泛应用, 新一代协同办公系统给传统办公自动化模式带来的变革必将掀起信息化建设领域的新高潮.

参考文献

1 朱焱.安徽烟草协同办公系统分析与设计.计算机与现代化,2009,7:147–150.

2 郑蓉,陆丽芳.基于SaaS的协同办公平台的架构设计与实现.计算机时代,2010,12:19–22.

3 康永,唐巍,程伟华.基于 SOA 架构的协同办公平台.计算机系统应用,2011,20(3):129–130,144.

4 何碧莲.基于 J2EE 平台协同办公系统的研究与设计.计算机与现代化,2010,8:188–190.

5 王艳芳.移动互联网技术下的协同办公管理变革.和田师范专科学校学报,2011,30(1):205–206.

6 泛微软件有限公司.泛微协同管理应用平台产品说明书.http://www.weaver.com.cn/.

7 王敏.企业信息化成熟度分类模型.福建电脑,2009,1:105,113.

8 刘昊江.协同办公系统在国有企业管理中的应用与探索.计算机光盘软件与应用,2011,18:5.

9 王嘉,王振宇,赵云丰.基于知识管理的协同办公系统的研究与应用.计算机技术与发展,2011,21(4):52–55.

10 段秀云.企业信息系统柔性多维度分析与评价[硕士学位论文].大连:大连理工大学,2008.