密文数据库查询优化技术①

李刚彪 闫宏印 (太原理工大学 计算机与软件学院 山西 太原 030024)

摘 要: 为了解决数据库加密后查询效率低的问题，针对数据库加密技术的实际应用，提出了一些解决方案。通过对数值型密文数据使用标志位，为字符型数据创建外存索引文本，缩小了查询范围，提高了数据查询效率。最后进行实验测试分析，证实了方法的有效性。

关键词: 数据库加密；标志位；索引；优化

Optimization Techniques for Query on Encrypted Databases

LI Gang-Biao, YAN Hong-Yin

(College of Computer Engineering and Software, Taiyuan University of Technology, Taiyuan 030024, China)

Abstract: In order to solve the problem of querying inefficiently on encrypted databases, some solutions are proposed in this paper based on practical application of database encryption techniques. Through setting marks for encrypted numerical data and creating index text in external memory, the range of querying is reduced, then the efficiency is improved. Finally, some tests are performed to validate the solutions.

Keywords: database encryption; marks; index; optimize

1 引言

互联网以及互联网上信息的应用是政府及企业最重要的战略性资产之一。数据库的安全防护尤为重要。数据加密技术通过对明文进行复杂的加密操作，以密文形式把数据存储起来，使得攻击者既使绕过操作系统或 DBMS(数据库管理系统)直接入侵到系统中，窃取了数据库中数据，也无法直接获得明文，从而在系统的存储层上保证了数据的安全性。但是数据加密后，失去了数据本身所固有的一些特性，如有序性，相似性和可比性，这样导致对加密数据的查询，往往需要对所有加密数据先进行解密，然后才能进行查询。而解密操作的代价往往很大，这样使得系统的性能急剧下降。

当前对密文数据的查询优化方法主要是针对数值型数据的加密，而对其它数据类型(如:字符型)的加密研究很少。本文针对数据库加密技术的实际应用，对数值型数据、字符型数据的查询优化，提出了一些解决方案。

2 相关的研究工作

为了能够提高加密数据的操作性能，研究人员采用不同的方法，提出了一些有效的措施，如秘密同态加密技术、过滤技术、索引技术等。

Hacigumus H,Lyer B 等[1]提出一种基于数据库-服务-提供者模型对加密数据进行查询的方法，该方法可以支持各种数据类型的精确匹配查询，但对于字符串数据的模糊匹配查询(如 like)无效；崔宾阁等[2]提出了为待加密的字段建立辅助索引字段的两阶段查询方法，能够实现各种类型的查询请求，同时系统的性能也得到了提高；王正飞等[3]提出一种对加密字符串数据进行模糊匹配查询的方法，可以有效实现对加密字符串数据的模糊匹配查询。但无法解决加密字符串数据的范围查询(如>、<)问题。

本文的研究建立在四个假设的前提下：1)假设数据库中有重要数据，需要加密保护。2)假设数据有可能被窃取。3)假设数据库表中数据量很大，有必要对查询做优化。4)假设加密密钥的存储管理是安全的。密钥的管理是数据加密技术的核心问题，本文不做讨论。

① 收稿时间:2009-12-02;收到修改稿时间:2010-01-19

方便论文叙述，我们先建立一个数据表(见表 1)。关系模式为： Item(NO，Name，Wages，CardID，Private1)；成员表 Item 的属性为员工号、姓名、工资、身份证号、个人信息，其中假定工资、身份证、个人信息是需要加密存储的。

表 1 成员表 Item

NO	Name	Wages	CardID	Private
20090001	张三	3200	1405211983091	张三的隐私
20090002	李四	4200	1405211983092	李四的隐私
20090003	王五	7000	1405211983093	王五的隐私
20090004	赵六	1000	1405211983094	赵六的隐私

3 数值型数据查询优化

数值型数据是数据库中使用最广泛的数据类型，对它的加密研究[4]也比较多。由于加密后的数值数据失去了本身固有的特性：有序性、可比性等，导致对一个数值密文的查询需要先对表中的整列数据进行解密操作，大大影响了查询效率。需要加密的数值型数据在数据表中的数据类型必须是文本类型，因为数值加密后的密文并不一定是数值型。对于数值型数据查询的优化，本文分两种情况进行讨论：等值查询和范围查询。

3.1 等值查询(“=”、“≠”)

如果查询运算符为“=”，则我们可以直接先将查询关键字(key)加密，然后再将关键字加密后的密文作为关键字进密文数据库查询。

例如，查询身份证号为 1405211983093 的员工记录。我们先将身份证号加密：

Encode(1405211983093)

然后执行查询：

SELECT * FROM ITEM WHERE CardID = Encode(1405211983093)，

执行完后，再用解密函数 Decode ()对查找到的记录中的密文字段进行解密操作，最后得到结果记录。Encode()为加密方法， Decode()为解密方法。“≠”运算符的查询类似。

3.2 范围查询（“>”、“<”、“BETWEEN”等=）

由于数值型数据加密后密文失去了可比性，若想恢复数值数据的可比性，就必须对数据先解密，再进行查询操作。例如：查询工资大于 3000 的员工记录，我们就必须先把工资(Wages)整列解密，然后再进行比较查询操作。如果表的数据量很大，查询效率无疑是很低的。我们可以采用缩小解密范围的方法对查询做优化。

一种方法是通过索引技术[5]提高对加密数据查询的性能，不过代价较大，且某些数值数据并不方便建立索引。针对实际应用，本文提出一种比较简便的方法，在数据表中添加一个标志位字段，通过标志位去缩小解密范围，再对记录进行解密操作，标志位的设置可以根据实际需求由编程人员在程序中设定方法计算得出。

例如,假设成员表中的工资项数据集中在 1000 到 10000 之间，小于 1000 和大于 10000 的记录比较少，那么我们就可以修改成员表的关系模式为：Item (NO，Name，Wages，CardID，Private，Mark)，标志位数据类型为数值型，范围为：1，2，3，...,10。则在插入成员记录时，设置标志位为：GetMark (wages)，GetMark( )为设置标志位函数(具体标志位取值如表 2)。

表 2 标志位设置

1	2	.....	9	10
<1000	1000~1999	.....	8000~8999	>10000

范围查询时，先调用 GetMark ()方法确定范围上下限的标志位，通过标志位去缩小解密范围，再对记录进行解密操作，比较得到结果记录。这个方法对等值查询也是适用的，先通过标志位缩小解密范围，再加密关键字，然后用关键字密文进行查询，对查到的记录解密，得到最终记录。

标志位的设计非常重要，上面例子中，如果窃取数据的人仅仅想知道员工工资的比较次序，那么，设置标志位反而造成了安全隐患。本文仅提出标志位缩小查询范围的思路，标志位的具体设定应该由数据库管理人员根据数据库数据特点来设计。去掉标志位的有序性或者用字符做标志位，且目标数据全是密文，无从参考，这样标志位方法的安全性是可以保证的，尤其对于某些数据库中不具有可比意义的数据(如银行卡账号等)，使用标志位方法提高查询效率会十分有意义。

当然，此方法对于 AVG、COUNT 等操作的查询是无效的，目前还没有针对这些查询的优化方法，要实现这些操作，仍须先解密表中整列密文数据。

4 字符型数据查询优化

字符型数据的等值查询与数值型的等值查询方法基本一样，不过单英文字符串数据的等值查询略有不

同，如果数据表中需加密的字段为单英文字符串(如账号、密码)，则可以考虑以 26 个英文字母(若字符串中有数值数据，则加上 0 – 9 十个数值)为索引字段建索引，缩小解密记录的范围。方法与数值数据做索引类似，本文不做详细讨论。

字符型密文数据查询的研究重点在于：模糊匹配查询(“LIKE“)。对字符型数据建立索引会相当复杂，可能会带来一些问题，导致在运行的过程中出错，得不偿失。本文的方法是在外存建一个类似于索引表的索引文本的方法去提高查询效率。方法步骤如下：

第一步，先建立一个文本文件(如:index.txt)。实际应用中，文件类型与存储位置由编程人员设定。

第二步，对数据加密以前，需先将加密的字段数据和主码数据保存到 index.txt 中。(保存格式见表 3)

表 3 保存格式

加密字段数据	|	主码	换行符

第三步，将加密数据插入数据库。同时将索引文本 index.txt 用文件加密算法(如 AES Rijnadel 算法)加密，这是很关键的一步。本文假设的前提是数据库数据可能被窃取，通过数据的加密来确保数据的安全，那样的话，本方法建立的加密外存文件并没有降低数据库数据的安全度。

当应用程序需要对密文数据做查询操作时，例如：从 Item 表查询个人信息中有“张三”字段的记录，则先将索引文本 index.txt 读入内存解密，然后用字符串匹配函数以“张三”为匹配字段对文本做匹配操作，通过匹配出的所有位置信息，取出对应行中“|”标识右边的主码信息，并将主码信息存到一个字符串数组 key[] 中去，然后通过各个主码数据去数据库中查询出对应的所有记录，解密数据，最后在应用程序中显示结果记录。

建立外存索引文本方法增加了对文件的加、解密操作，以及对文件的字符串匹配查询。首先，在内存中对字符串匹配操作比用 SQL 语句去数据库中查询的效率高得多，其次，执行一次查询对文件只需要一次解密操作，而若直接进数据库查询密文记录，则有多少条记录就需解密多少次，效率的提升是可以肯定的(具体见效率测试)。

5 效率测试

测试目的：验证本文方法在优化查询效率方面的有效性。数值型数据的优化效果是显而易见的，所以本文着重对文本型数据的查询优化做测试分析。

测试环境：

操作系统：Windows Vista 32 位系统；

处理器：Intel T6400 2.0GHZ；

内存：2G；

程序语言：Java；

IDE：MyEclipse 7.5；

DataBase：MySQL 5.0.41；

加密算法：AES Rijnadel 算法；

加密密钥：“这是一个测试加密密钥”；

效率测试工具：JUnit。(一个 java 源码开发的测试框架，测代码执行时间)。

测试步骤：

1)用 JAVA 语言实现 AES Rijnadel 加密算法类 AESTest。

2)创建一个文本文件 index.txt，并且在数据库 test 中创建一个数据表 Test，Test 表中只有一列，列名为：cipherText。

3)用 AESTest 类的加密方法 Encode（）加密“张三的隐私|20090001”字符串，得到密文 cipher(“0Q2Toc/nqpoxtycutdUh+bkw3qPv8nJ3OvSwoW2P+NI=”)，做循环程序分别在 index.txt 和数据表 Test 中插入 10000 行密文数据 cipher。

4)写一个测试类 TestAES，包含两个测试方法 Test1()，Test2()，部分主要代码如下：

Test1:(外存索引文本法)

```
BufferedReader br=new BufferedReader(
                new InputStreamReader(
                  new
    FileInputStream("index.txt")));
        long m = 0; String clearText = null;
        String strKey = "这是一个测试加密密钥";
        final int BLOCK_SIZE = 24;//加密分块值
         while ((clearText = br.readLine()) != null) {
            if(Rijndael_Util.decode(strKey, clearText,
                BLOCK_SIZE).indexOf("张三") != -1)
```

```
                                        //解密数据
                    m++;  //记录匹配到的次数
            }
              System.out.println(m);

        Test2：（先解密所有记录后查询）
            ResultSet result = null; long m = 0;
        Class.forName("org.gjt.mm.mysql.Driver");

        con = DriverManager.getConnection(
         "jdbc:mysql://localhost:3306/Test",
            "root", "sa");  //连接数据库

            stat = con.prepareStatement("select *
from test");
            result = stat.executeQuery();  //执行
SQL
            while (result.next()) {
                String row = result.getString(1);
                if (row != null)
                    if (Rijndael_Util.decode(strKey,
                     row, BLOCK_SIZE).indexOf("张三
") != -1)
                m++;
            }
            System.out.println(m);
```

测试结果：用 JUnit 执行测试类，执行 10 次，时间测试结果见表 4。

表 4　测试时间（时间单位：s）

	1	2	3	4	5
Test1	0.150	0.142	0.161	0.154	0.151
Test2	0.703	0.701	0.711	0.701	0.717
	6	7	8	9	10
Test1	0.150	0.148	0.149	0.150	0.151
Test2	0.711	0.708	0.703	0.695	0.702

Test1 的平均时间≈0.151 s　Test2 的平均时间 ≈ 0.705 s

Test2 的平均时间/ Test1 的平均时间 ≈ 4.67

实验数据表明，使用创建外存索引文本的方法，对字符型数据模糊匹配查询效率的提升是非常明显的。在实际操作中，我们可以考虑分类建一些索引文本，如分为中文索引文本和英文索引文本，应用程序判断查找关键字是中文则调用中文索引文本，若是英文则调用英文索引文本，分类更细致，性能的提升将会更为可观。

6　结论

本文针对数据库加密技术的实际应用需求，提出了一些数据库查询效率的优化方案。通过对数值型数据加标志位以及对字符文本数据建立外存索引文本，极大地提高查询效率，并通过编程实验测试，证明了方法的有效性。然而对于数据库的优化，放之四海而皆准的解决方案不太可能存在，只有通过分析自身加密数据库的数据特点，针对性地优化数据库。优化往往会降低数据库的安全度，当然任何数据库都不可能保证绝对的安全，尽可能的做到安全与效率综合最优，便是我们应该追求的。

参考文献

1 Hacigumus H, Lyer B, Li C, et al. Executing SQL over encrypted data in the database-server-provider model. The 2002 ACM SIGMOD Int'l Conf on Management of Data, Madison, Wisconsin ,2002.

2 崔宾阁,刘大昕,王桐等.支持快速查询的数据库加密方法研究. 微计算机信息, 2008,33 (6): 115－118.

3 王正飞,王曼,汪卫等.数据库中加密字符数据的存储与查询.计算机研究与发展, 2004,41 (suppl):66－71.

4 宋俊洪,汪冰等. 密文数据库检索方法的研究. 广东工业大学学报, 2007,24(2):93－95.

5 刘闪,袁丁.基于密文数据库中数组检索的研究.计算机应用, 2005,25(21):131－133.