

SQL Server 查询优化器原理与优化实例分析

刘维学

(渤海大学 信息科学与技术学院, 辽宁 锦州 121013)

摘要: 查询是数据库的核心操作,随着数据库技术的发展以及数据量急剧增加,对查询性能的要求越来越高,查询优化成为数据库管理系统亟待解决的重要问题。文中针对应用最广泛的 SQL Server 数据库的查询优化器进行研究。通过图形研究查询优化器的工作原理,并深入分析提交 SQL 语句、解析、代数化、查询优化、编译、执行、结果等查询优化器的工作步骤;进行实例分析,运用图形表示了逻辑树和经过优化后得到的查询执行计划。结果表明,SQL 语句是查询优化的基础,实际应用时需要写出符合查询优化器规则的 SQL 语句。

关键词: SQL Server;数据库;查询优化;查询优化器;查询执行计划

中图分类号:TP311

文献标识码:A

文章编号:1673-629X(2013)11-0108-04

doi:10.3969/j.issn.1673-629X.2013.11.027

Query Optimization Principle and Optimized Instance Analysis of SQL Server

LIU Wei-xue

(College of Information Science and Technology, Bohai University, Jinzhou 121013, China)

Abstract: The query is the core operation of the database, with the development of database technology and the dramatic increasing in amount of data, increasingly high demand on the query performance, query optimization becomes important issue that needs to be solved of database management system. It studies the query optimizer that the most widely used in SQL server database. Above all, research query optimizer's working principle through graph, penetrate deeply analysis of query optimizer's work steps by submitting SQL statements, parsing, algebrization, query optimization, compile, query execution, query results and so on; then, execute instance analysis, use graphics for representation of the logic tree and optimized query execution plan. The result shows that the SQL statements are the basis of the query optimization, practical application needs to write SQL statements that match the query optimizer rules.

Key words: SQL Server; database; query optimization; query optimizer; query execution plan

0 引言

数据库系统性能优化的基本原则是通过尽可能少的磁盘访问获得所需要的数据,从而提高系统吞吐量,降低用户响应时间,提高数据库的可用性。SQL (Structured Query Language), 中文称为结构化查询语言,是一种数据库查询和程序设计语言,用于存取数据以及查询、更新和管理关系数据库系统^[1]。SQL 集数据库模式定义语言 (DDL, Data Definition Language)、数据查询语言 (DQL, Data Query Language)、数据操纵语言 (DML, Data Manipulation Language)、数据库控制语言 (DCL, Data Control Language) 功能于一体,可以独立完成数据库管理系统 (DBMS, DataBase Manage-

ment System) 的全部活动,且具有良好的可扩展性。在 SQL 中应用最多的是数据查询语言,SQL 对数据库的性能影响很大,数据查询语言是影响数据库查询性能乃至数据库性能的决定性因素。

查询优化在查询处理阶段对于选择执行查询的最有效策略起着至关重要的作用^[2]。当提交一条 SQL 语句时,DBMS 进行语法检查后,将语句提交给查询优化器,优化器再将 SQL 语句按照一定的优化方法分析为各个组成。用户提交的 SQL 语句是系统优化的基础,一个不合理的查询计划仅通过查询优化器进行优化,不可能高效,因此 SQL 语句书写的优劣至关重要。文中通过研究查询优化器的工作原理来启发并指导用

户写出较优的 SQL 语句^[3]。

1 查询优化器原理

查询优化器 (Query Optimizer) 是负责生成 SQL 语句有效查询执行计划 (QEP, Query Execution Plan) 的 SQL Server 数据库引擎组件。具体来说,查询优化器接收提交给 SQL Server 的 SQL 查询语句,考察解决查询各个部分 (如条件、连接和函数等) 的多种可能方法,用穷举法列出查询的所有执行策略,通过估算每个操作的每项逻辑开销,并考虑可用的索引、硬件的限制和数据的统计信息,从而产生最佳执行计划,然后执行计划,并提供所需的结果^[4]。SQL Server 查询优化器的工作原理如图 1 所示。

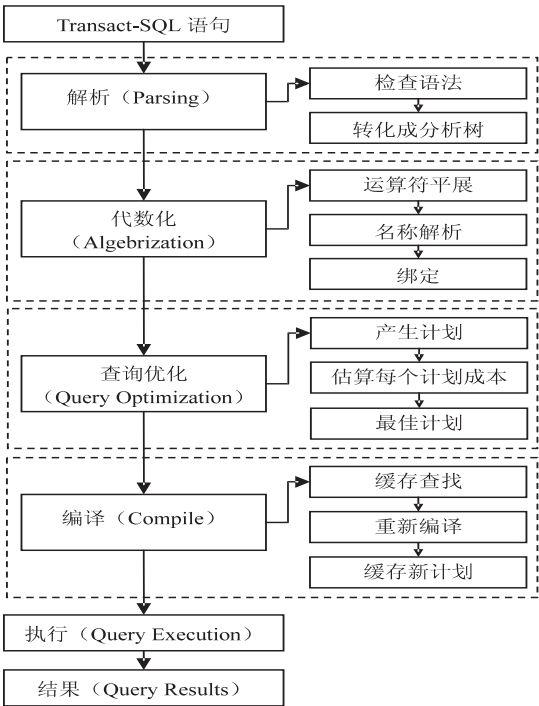


图 1 查询优化器工作过程

第 1 步:提交 SQL 语句。可以是标准的 SQL 语句,也可以是扩展的 Transact-SQL 语句,还可以是批处理 (作为一个单元编译的一个或多个 T-SQL 语句的组合,存储过程就是一个批处理的例子)。

第 2 步:解析处理。为传来的 SQL 语句检查语法是否存在错误,并将语法分解成能被关系数据库引擎响应的组件部分,最后输出解析好的查询结果。解析结果是一个逻辑树,逻辑树中的每一个节点都表示这个查询进行的每个操作。

第 3 步:代数化。使用 Algebrizer 来完成,主要功能是绑定,因此代数化过程通常称为绑定。将上一步的分析树作为输入,生成被称为查询处理器树的输出,用于查询优化。主要包括三项工作:

●运算符平展。简单的讲就是把二元运算符

UNION、AND 和 OR 等组合成 N 元运算符,从而避免递归过程。在分析阶段,分析器把逻辑运算符都看作是二元的,如图 2(a) 所示,平展后如图 2(b) 所示^[5]。

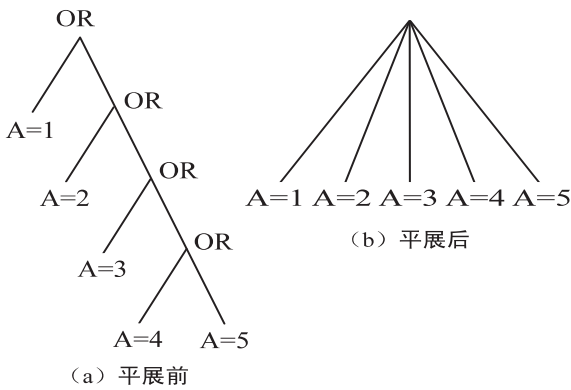


图 2 运算符平展

●名称解析。逻辑树中的每个表名称和列名称都关联到相应表或列对象的引用。名称解析检查查询中的每个对象名称是否真的引用了系统目录中已经存在的有效表或列,以及是否在特定的查询范围内可见。然后,把目录中的信息关联到该对象的名称。其中,对视图的名称解析使用视图树替换视图引用的过程。如果视图还引用了其他视图,将被递归地进行解析。

●绑定。进行聚合绑定和组分绑定,经过绑定后,就会产生另外一种树形的数据结构,也称为查询处理器树,传递给下一个步骤。

第 4 步:查询优化。从几个可行的方案中选出一个执行计划的过程称作优化。查询中以何种顺序访问表,使用哪种方法和使用哪个索引,由哪个连接算法等都是由查询优化器组件来完成的,但是这个选择过程不是随意的,必须满足的前提条件是保证最后得到的结果是正确的^[6]。查询优化主要过程如下:

●产生计划。查询以 Transact-SQL 的形式提交给 SQL Server。因为 SQL 语句是一个高层抽象的声明语言,只是定义了要从数据库中获取什么数据,而没有定义如何获取数据,即没有定义获取数据的方法和步骤。所以,对于 SQL Server 所接收到的每一个查询,查询处理器的首要任务就是产生查询计划,即多个候选技术,用来描述如何执行查询。

●估算每个计划成本。一个逻辑操作可以有很多的物理操作与其对应,而每个物理操作的成本不一样,同时,也没用所谓的“什么物理操作比其他的物理操作更优”,一切视情况可认定。因此,查询优化器会综合考虑很多的情况估算每个计划的成本,即执行每个计划的开销。

●最佳计划。当一个查询被传递到数据库系统时,优化器需要评估不同查询计划的成本和查询访问路径选择^[7]。通过估算多个候选计划的成本,从中选

择一个“比较优”的查询计划,传递给存储引擎。

第 5 步:编译。查询被编译成为机器可执行的代码。SQL Server 有一个用来存储执行计划和数据缓存的内存池。根据系统状态,执行计划和缓存所占内存池的百分比是动态的。内存池中存储执行计划的部分叫做过程缓存^[8]。执行计划块是一种可重新再用、能够被任意多个用户使用的只读数据结构。在内存中,从来不会有多于两个的执行计划副本:用于串行计划和并行计划的副本。每个执行查询的用户都有一个数据结构,这个数据结构包含有指定给执行动作的数据,例如参数值。当执行一个 Transact-SQL 语句时,SQL Server 扫描过程缓存区来决定是否存在一个相同的 Transact-SQL 语句的执行计划。如果存在这个计划,SQL Server 重用这个计划,节省了 Transact-SQL 语句重新编译的开销。如不存在这个计划,SQL Server 为这个查询产生一个新的执行计划。如数据发生了某种变化,导致了内存中的执行计划不再准确时,SQL Server 将使执行计划无效。系统为执行查询的下一个查询重新编译。导致执行计划无效的条件包括:更改了表或视图的结构,删除了执行计划使用的某个索引,大量地修改键值或者对查询引用的表执行了 INSERT、UPDATE、DELETE 语句等。编译过程如图 3 所示。在 ALTER DATABASE 语句中将 PARAMETERIZATION 选项设置为 FORCED 可启用强制参数化。强制参数化通过降低查询编译和重新编译的频率,可提高数据库的性能。能够通过强制参数化受益的数据库通常是需要处理来自源的大量并发查询的数据库。

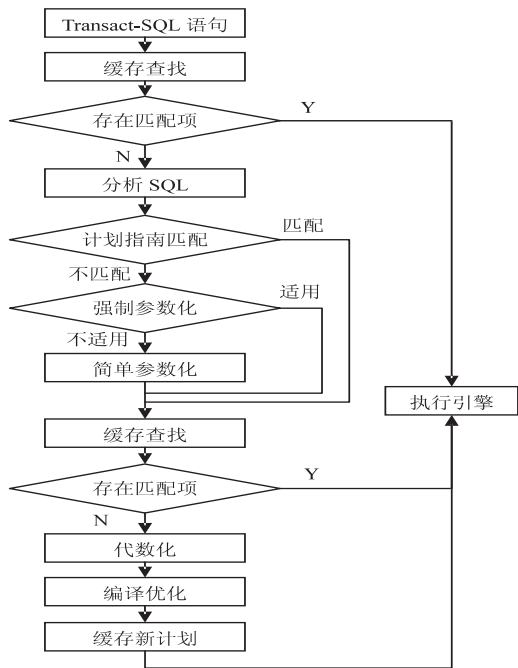


图 3 编译过程

第 6 步,执行。查询优化器通过执行表扫描,或使用可用的索引,决定访问数据的最好方法,然后运用最好方法执行查询。

第 7 步,结果。得到查询结果,供应用程序或其他程序使用。

2 查询优化实例分析

优化是查询处理中一个最复杂也是最重要的部分。分析器和 Algebrizer 创建的逻辑树是优化器的输入。优化器需要逻辑树、与查询所涉及的对象(例如表、索引、统计信息和约束等)有关的元数据和硬件信息。优化器使用这些信息来创建编译计划,它由物理运算符组成。逻辑树包括一些逻辑运算符,用来描述做什么,如“读取表”、“连接”等。优化器生成的物理运算符指定算法,描述如何做,如“索引查找”、“索引扫描”、“哈希连接”等。优化器告诉 SQL Server 如何执行这些步骤,可高效地获得结果^[9]。以“学生-课程数据库”为例进行实例分析,该数据库包括三个表:

● 学生表,由学号、姓名、性别、年龄、专业等五个字段组成,表名称及属性名称如下:

Student(Sno, Sname, Ssex, Sage, Sprof)

其中“Sno”为主关键字。

● 课程表,由课程号、课程名、学分等三个字段组成,表名称及属性名称如下:

Course(Cno, Cname, Ccredit)

其中“Cno”为主关键字。

● 学生选课表,由学号、课程号、考试成绩等三个字段组成,表名称及属性名称如下:

SC(Sno, Cno, Grade)

其中“Sno、Cno”共同构成主关键字。

在该数据库中,学生表与课程表是多对多的关系,即一名学生可以选修多门课程,一门课程可以被多名学生所选。学生选课表是多对多关系转换的中间表,因此,学生表和学生选课表是一对多的关系,Sno 是外关键字;课程表和学生选课表是一对多的关系,Cno 是外关键字。

现将“软件工程”专业学生的平均成绩按学号排序,SQL 语句如下:

```
SELECT S. Sno, Agrade = AVG( SC. Grade)
FROM Student S, Course C, SC
WHERE S. Sno = SC. Sno and SC. Cno = C. Cno and S. Sprof =
'软件工程'
GROUP BY S. Sno
ORDER BY S. Sno
```

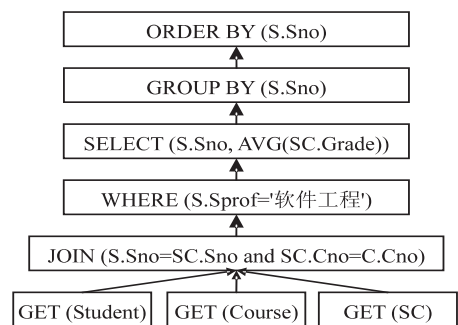
在企业管理器中执行后,SQL 语句规范为内连接的形式,如下:

```

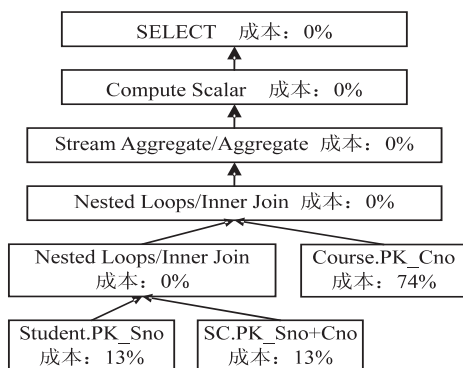
SELECT S. Sno, AVG(SC. Grade) AS Agrade
FROM Student S INNER JOIN
      SC ON S. Sno = SC. Sno INNER JOIN
      Course C ON SC. Cno = C. Cno
WHERE (S. Sprof = '软件工程')
GROUP BY S. Sno
ORDER BY S. Sno

```

进行解析和绑定时,完成了一棵由逻辑关系操作符组成的树,如图 4(a)所示。查询优化器经过优化后得到的查询执行计划如图 4(b)所示。



(a) 逻辑树



(b) 执行计划

图 4 查询优化示例

优化器通过更换逻辑运算为它所知道的对应的物理操作来生成整个执行计划。这种类型转换是由优化器内部通过实施一系列规则来完成的。如逻辑运算“INNER JOIN”,对应的物理操作则是嵌套循环(Nested Loops),还存在 merge 和 hash join 规则^[10]。

所展示的从逻辑树到执行计划的图中,优化器共匹配和执行了 5 个规则:

- (1) GET 转化为扫描(scan);
- (2) Join 转化为嵌套循环(nested loop);
- (3) SELECT 转化为过滤器(Filter);
- (4) Group by Aggregate 转化为 stream aggregate;
- (5) Group by Aggregate 转化为 Hash aggregate。

3 结束语

数据查询语言是一种描述性的语言,在 SQL 语句中只需要描述问题,SQL Server 会确定如何来执行查

询。查询优化器的主要任务是利用已有的关于索引和列数据的统计信息选择用于执行查询和更新操作的最有效的方案^[11]。一个 SQL 语句在 DBMS 中可有很多条执行路径,虽然执行这些路径返回的结果完全相同,但是执行所花费的代价却有很大差异,而查询优化器的作用就是找出那条花费代价最小的路径。连接的选择对查询性能影响较大,因此查询优化器必须能够通过一定的算法确定一个好的连接次序,以便对查询路径进行优化。查询优化器通常是基于全局数据统计来生成单一查询计划^[12-14],实际应用中的数据集市往往具有非均匀分布性,选择单一查询计划可能会导致大部分无效的查询执行,这是需要进一步深入研究的问题。

参考文献:

- [1] 百度百科. 结构化查询语言[EB/OL]. 2013-01-10. <http://baike.baidu.com/view/34.htm>.
- [2] Tanian D, Khaw H Y, Tjioe H C, et al. The use of Hints in SQL-Nested query optimization[J]. Information sciences, 2007, 177(12): 2493-2521.
- [3] 李 菲. SQL Server 数据库查询优化方法探究[J]. 福建电脑, 2008, 24(7): 76-77.
- [4] AgileSharp. 浅析 SQL Server 查询优化器的工作原理[EB/OL]. 2013-01-10. <http://tech.it168.com/a2012/0329/1332/000001332016.shtml>.
- [5] YinWangLive. TSQL 查询内幕:(2.2)编译[EB/OL]. 2013-01-10. <http://tech.ddvip.com/2009-04/1239706028115034.html>.
- [6] 4inwork. 教你如何优化 SQL 语句中的物理查询[EB/OL]. 2013-01-10. <http://database.51cto.com/art/200904/121727.htm>.
- [7] Kao Kuo-Fong, Liao I-En. An index selection method without repeated optimizer estimations[J]. Information sciences, 2009, 179(13): 2263-2272.
- [8] 梁方明. SQL Server 2000 数据库编程[M]. 北京:北京希望电子出版社, 2002.
- [9] Knight B, Patel K. Professional Microsoft SQL Server 2008 Administration[M]. USA: Wiley Publishing, 2009.
- [10] Kinzent. 深入理解 SQL Server 查询优化器-构造执行计划(PART I)[EB/OL]. 2012-12-18. <http://bbs.csdn.net/topics/390320092>.
- [11] 石剑平, 蔡光程. SQL Server 2005 查询优化技术的研究与实现[J]. 信息系统工程, 2010, 23(5): 78-79.
- [12] 孙振兴, 向 阳, 刘增宝. PostgreSQL 查询优化器分析研究[J]. 计算机技术与发展, 2011, 21(8): 141-144.
- [13] 王 力, 王成良. 基于免疫遗传算法的关系型数据库查询优化技术[J]. 计算机系统应用, 2008, 18(1): 72-75.
- [14] Nehme R V, Works K, Lei C, et al. Multi-route query processing and optimization[J]. Journal of computer and system sciences, 2013, 79(3): 312-329.

SQL Server查询优化器原理与优化实例分析

作者：[刘维学](#)，[LIU Wei-xue](#)
作者单位：[渤海大学 信息科学与技术学院, 辽宁 锦州, 121013](#)
刊名：[计算机技术与发展](#)

ISTIC

英文刊名：[Computer Technology and Development](#)

年，卷(期)：2013(11)

本文链接：http://d.g.wanfangdata.com.cn/Periodical_wjfz201311028.aspx