POSTGRESQL中并行查询代价估算方法

作者：Hans-Jürgen Schönig 译者: 陈雁飞 崔鹏

### 作者简介

**Hans-Jürgen Schönig**：从20世纪90年代开始使用PostgreSQL，担任CYBERTEC公司的CEO和技术主管（[www.cybertec-postgresql.com](http://www.cybertec-postgresql.com)）， CYBERTEC是该领域的市场领导者之一，自2000年以来已为全球无数客户提供服务。

### 译者简介

**陈雁飞**：开源PostgreSQL爱好者，一直从事PostgreSQL数据库运维工作

**崔鹏：**任职于海能达通信股份有限公司，致力于PostgreSQL数据库在专网通信领域的应用与推广

v 在PostgresSQL9.6中引入了并行查询，从那之后这一功能不断地得到扩展。在PostgreSQL11和PostgreSQL12中，数据库引擎增加了更多的功能。然而，仍然有一些与并行查询相关的问题，这些问题经常出现在培训过程中，因此有必要澄清说明一下。

## 顺序扫描的代价估计

为了说明并行查询的工作原理，创建一个简单的表，仅仅有两列，如下：

|  |
| --- |
| test=# CREATE TABLE t\_test AS    SELECT id AS many, id % 2 AS few    FROM generate\_series(1, 10000000) AS id;  SELECT 10000000    test=# ANALYZE;  ANALYZE |

列many包含有1000万条不同数据，列few包含有两个不同的值。但是，出于演示用例的考虑，所有合理的大表都可以使用。

我们用来分析PostgreSQL优化器如何工作的查询语句也非常简单，如下：

|  |
| --- |
| test=# SET max\_parallel\_workers\_per\_gather TO 0;  SET  test=# explain SELECT count(\*) FROM t\_test;                                   QUERY PLAN  ------------------------------------------------------------------------   Aggregate (cost=169248.60..169248.61 rows=1 width=8)     -> Seq Scan on t\_test (cost=0.00..144248.48 rows=10000048 width=0)   (2 rows) |

默认配置将自动使PostgreSQL进行并行顺序扫描，为了更加方便阅读，我们希望先不使用并行查询。

通过设置max\_parallel\_workers\_per\_gather为0可以关闭并行查询。如在执行计划中所看到的，顺序扫描的估算成本为144248，而整个语句估算成本是169248。那么PostgreSQL是如何计算得到这个数字呢？让我们先看下如下详细内容：

|  |
| --- |
| test=# SELECT pg\_relation\_size('t\_test') AS size,                pg\_relation\_size('t\_test') / 8192 AS blocks;    size     | blocks  -----------+--------   362479616 | 44248  (1 row) |

表t\_test大概占用350MB空间并包含有44248个物理块。每个块将被顺序地读和处理。必须对这些块中所有行统计完才能得到最终结果。优化器使用下面的公式计算代价：

|  |
| --- |
| test=# SELECT current\_setting('seq\_page\_cost')::numeric \* 44248              + current\_setting('cpu\_tuple\_cost')::numeric \* 10000000              + current\_setting('cpu\_operator\_cost')::numeric \* 10000000;      ?column?  -------------   169248.0000  (1 row) |

可以看到，这里使用了两个参数：优化器使用seq\_page\_cost表示顺序读取一个块的代价。更重要的是，必须考虑到一个事实，即所有这些行在最终统计之前还必须通过CPU（cpu\_tuple\_cost）。使用cpu\_operator\_cost是因为计数基本上和为每一行调用“+1”相同。因此，在执行计划中得到顺序扫描的总代价为169248。

## 并行顺序扫描估计

在Cybertec数据库培训期间，有很多人对PostgreSQL的顺序扫描估算方式感到困惑并对这个有很多疑问。首先看下面的执行计划，看看会发生什么：

|  |
| --- |
| test=# SET max\_parallel\_workers\_per\_gather TO default;  SET  test=# explain SELECT count(\*) FROM t\_test;                                    QUERY PLAN  ------------------------------------------------------------------------------------   Finalize Aggregate (cost=97331.80..97331.81 rows=1 width=8)   -> Gather (cost=97331.58..97331.79 rows=2 width=8)      Workers Planned: 2      -> Partial Aggregate (cost=96331.58..96331.59 rows=1 width=8)         -> Parallel Seq Scan on t\_test (cost=0.00..85914.87 rows=4166687 width=0)  (5 rows) |

如看到的那样，PostgreSQL决定使用2个CPU。但是PostgreSQL是如何考虑到这一点的呢？“rows=4166687”

答案就在下面的公式中：

10000048.0 / (2 + (1 – 0.3 \* 2)) = 4166686.66 rows

PostgreSQL期望表中的行数为10000048（由前一次ANALYZE分析决定）。接下来的事情就是PostgreSQL试图决定一个核能做多少工作。但是上面这个公式的具体含义又是什么呢？

estimate = estimated\_rows / (number\_of\_cores + (1 – leader\_contribution \* number\_of\_cores)

|  |
| --- |
| How-Postgres-estimates-parallel-queries |

Leader进程通常需要花费大量精力给最终的结果。但是，假设leader花费大约30%的时间为工作进程提供服务。因此，随着核心数的增加，leader的贡献将降低。如果有4个或者更多的核工作，那么leader将不再对扫描有任何有意义的贡献——因此，PostgreSQL只会根据核心数来计算表的大小，而不是使用上面的公式。

## 其他并行操作

其他并行操作将使用类似的除数来估计这些操作所需的工作量。位图扫描等工作方式相同。

## 原文地址

<https://www.cybertec-postgresql.com/en/how-postgresql-estimates-parallel-queries/>