实验6：查询计划执行

**1 实验目的**

1. 学习并掌握数据库管理系统中查询处理与查询优化的基本知识和方法。
2. 通过查询计划执行工具（explain）体会不同优化策略对查询执行性能的影响，掌握基本的查询优化方法。

**2 实验平台与工具**

1. Windows、Linux操作系统
2. 数据库管理系统PostgreSQL

**3 实验内容与要求**

利用PG中的explain工具，对比不同连接算法在SQL查询语句执行中的作用。

1. 针对Nested Loop Join，Hash Join，Merge Sort Join三种连接进行比较，可以选择其中两种，也可以选择全部三种参与实验，自行决定。
2. 实验采用的数据库及其基本表自行决定。
3. 通过实验展现不同情况下（包括大小表、有无索引、不同的连接比较条件）物理优化的效果，对实验步骤及其结果进行必要说明和解释（包括时间代价、存取路径的策略等），最终实现实验目的2的达成。

**4. 要求**

1. 独立完成，严禁相互抄袭（如有发现抄袭和被抄袭均判为0分），以及从网络上直接摘抄别人的观点和总结（该行为将影响报告成绩）。
2. 实验报告符合学术写作的排版要求，请参考群文件中的“报告模板.docx”和“参考文献格式.docx”的排版格式。
3. 实验报告内容详实，采用图文混合的方式叙述安装和配置过程。

Tip：Win+Shift+S 在Windows中可以快速截屏。

1. 报告文件见附件，提交报告时请以附件形式插入到超星作业中。

实验报告

报告标题：查询计划执行

学号：19200312

姓名：成文涛

日期：2022.12.29

# 一、实验环境

1. 操作系统：win10

2. 数据库管理系统软件（含版本号）：pgadmin4

# 实验内容及其完成情况

我们有一个嵌套循环连接节点，其中包含两个表扫描作为输入或子项。

1. EXPLAIN ANALYZE **SELECT** \*
2. **FROM** tb1 t1, tb2 t2
3. **WHERE** t1.unique1 < 10 AND t1.unique2 = t2.unique2;
5. QUERY PLAN
6. ---------------------------------------------------------------------------------------------------------------------------------
7. Nested Loop  (cost=4.65..118.62 **rows**=10 width=488) (actual **time**=0.128..0.377 **rows**=10 loops=1)
8. ->  Bitmap Heap Scan **on** tb1 t1  (cost=4.36..39.47 **rows**=10 width=244) (actual **time**=0.057..0.121 **rows**=10 loops=1)
9. Recheck Cond: (unique1 < 10)
10. ->  Bitmap **Index** Scan **on** tb1\_unique1  (cost=0.00..4.36 **rows**=10 width=0) (actual **time**=0.024..0.024 **rows**=10 loops=1)
11. **Index** Cond: (unique1 < 10)
12. ->  **Index** Scan using tb2\_unique2 **on** tenk2 t2  (cost=0.29..7.91 **rows**=1 width=244) (actual **time**=0.021..0.022 **rows**=1 loops=10)
13. **Index** Cond: (unique2 = t1.unique2)
14. Planning **time**: 0.181 ms
15. Execution **time**: 0.501 ms

计划程序选择使用哈希连接，其中将一个表的行输入到内存中的哈希表中，然后扫描另一个表并探测哈希表是否与每行匹配。再次注意缩进如何反映计划结构：位图扫描是 Hash 节点的输入，该节点构造哈希表。然后返回到“哈希联接”节点，该节点从其外部子计划中读取行，并在哈希表中搜索每个行。

1. EXPLAIN **SELECT** \*
2. **FROM** tb1 t1, tb2 t2
3. **WHERE** t1.unique1 < 100 AND t1.unique2 = t2.unique2;
5. QUERY PLAN
6. ------------------------------------------------------------------------------------------
7. Hash Join  (cost=230.47..713.98 **rows**=101 width=488)
8. Hash Cond: (t2.unique2 = t1.unique2)
9. ->  Seq Scan **on** tenk2 t2  (cost=0.00..445.00 **rows**=10000 width=244)
10. ->  Hash  (cost=229.20..229.20 **rows**=101 width=244)
11. ->  Bitmap Heap Scan **on** tb1 t1  (cost=5.07..229.20 **rows**=101 width=244)
12. Recheck Cond: (unique1 < 100)
13. ->  Bitmap **Index** Scan **on** tenk1\_unique1  (cost=0.00..5.04 **rows**=101 width=0)
14. **Index** Cond: (unique1 < 100)

合并联接要求其输入数据在联接键上排序。在此计划中，数据通过使用索引扫描以正确的顺序访问行来排序，但首选顺序扫描和排序，因为该表中还有更多要访问的行。（顺序扫描和排序经常击败索引扫描对许多行进行排序，因为索引扫描需要非顺序磁盘访问。

1. EXPLAIN **SELECT** \*
2. **FROM** tb1 t1, tb2 t2
3. **WHERE** t1.unique1 < 100 AND t1.unique2 = t2.unique2;
5. QUERY PLAN
6. ------------------------------------------------------------------------------------------
7. Merge Join  (cost=198.11..268.19 **rows**=10 width=488)
8. Merge Cond: (t1.unique2 = t2.unique2)
9. ->  **Index** Scan using tb1\_unique2 **on** tb1 t1  (cost=0.29..656.28 **rows**=101 width=244)
10. Filter: (unique1 < 100)
11. ->  Sort  (cost=197.83..200.33 **rows**=1000 width=244)
12. Sort **Key**: t2.unique2
13. ->  Seq Scan **on** onek t2  (cost=0.00..148.00 **rows**=1000 width=244)

# 三、实验总结

（可以总结实验中出现的问题以及解决的思路，也可以列出没有解决的问题）

问题1：

1. 问题描述

实验操作不难，但是做完很茫然，什么也看不懂，也总结不出什么，这就是内部的细节对用户封装吧，突然感受到数据内模式的复杂性，脑子的固化印象不能再是一张张的表格。

1. 问题分析（可能的原因、难点、挑战）

暂无。

1. 解决方案

（如该问题没有被解决可以不写具体的解决方案，而给出大致的思路和方向）

暂无。