第 10 讲:排序和聚合算法

15-445/645 数据库系统(2022 年秋季)

https://15445.courses.cs.cmu.edu/fall2022/

卡内基梅隆大学安迪·帕夫洛

1 排序

dbms 需要对数据进行排序，因为表中的元组在关系模型下没有特定的顺序。排序(可能)用于 ORDER BY、GROUP BY、JOIN 和 DISTINCT 操作符。如果需要排序的数据适合内存，那么 DBMS 可以使用标准排序算法(例如，快速排序)。如果数据不适合，那么 DBMS 需要使用外部排序，这种排序能够根据需要溢出到磁盘，并且更喜欢顺序而不是随机 I/O。

如果查询包含带有 LIMIT 的 ORDER BY，那么 DBMS 只需要扫描一次数据就可以找到 top-N 元素。这被称为 **Top-N 堆排序**。堆排序的理想场景是当 top-N 元素适合内存时，这样 DBMS 在扫描数据时只需要维护内存中排序的优先级队列。

对于大到内存容纳不了的数据进行排序的标准算法是**外部归并排序**。它是一种分治排序算法，将数据集分割成单独的*运行*，然后分别对它们进行排序。它可以根据需要将运行数据溢出到磁盘，然后一次读取一个运行数据。该算法由两个阶段组成:

阶段#1 -排序:首先，算法对适合主内存的小块数据进行排序，然后将排序后的页写回磁盘。

阶段#2 -合并:然后，算法将排序后的子文件合并成一个更大的单个文件。

双向归并排序

该算法最基本的版本是双向归并排序。该算法在排序阶段读取每一页，对其进行排序，并将排序后的版本写回磁盘。然后，在合并阶段，它使用三个缓冲页。它从磁盘中读取两个排序过的页，并将它们合并成第三个缓冲页。每当第三页填满时，它就被写回磁盘，并替换为一个空页。每一组排好序的页称为一*次运行*。然后该算法递归地将这些运行合并在一起。

如果 N 是数据页的总数，算法使 1 +⌈log 2N⌉total 通过数据(1 为第一个排序步骤，然后⌈log 2N⌉为递归合并)。总的 I/O 成本是 2N ×(# of passes)，因为每一遍执行一个 I/O 读和一个 I/O 写为每个页面。

通用(k 路)归并排序

该算法的通用版本允许 DBMS 利用使用三个以上缓冲页的优势。设 B 为可用缓冲页的总数。那么，在排序阶段，算法可以一次读取 B 页，并将 $N^{?}$ 个排序后的运行写回磁盘。合并阶段也可以合并到

B

B−1 在每次遍历中运行，同样使用一个缓冲页来保存合并的数据，并在需要时写回磁盘。

在广义版本中，算法执行 1 +logB-1?N ??通过(1 次用于排序阶段，B

?logB-1?BN？？为合并阶段。然后，总的 I/O 成本是 2N ×(# of passes)，因为它再次必须在每个 pass 中对每个页面进行读写。

双缓冲优化

外部归并排序的一种优化是在后台预取下一次运行，并在系统处理当前运行时将其存储在第二个缓冲区中。通过不断利用磁盘，这减少了每一步 I/O 请求的等待时间。这种优化需要使用多线程，因为预取应该在当前运行的计算发生的同时发生。

用 B +树

对于 DBMS 来说，有时使用现有的 B+树索引来帮助排序比使用外部合并排序算法更有利。特别是，如果索引是聚集索引，DBMS 可以只遍历 B+树。由于索引是聚簇的，数据将按照正确的顺序存储，因此 I/O 访问将是顺序的。这意味着它总是比外部归并排序好，因为不需要计算。另一方面，如果索引是非聚类的，遍历树几乎总是更糟糕，因为每个记录都可以存储在任何页面中，因此几乎所有的记录访问都需要读取磁盘。

2 聚合

查询计划中的聚合操作符将一个或多个元组的值折叠为单个标量值。有两种实现聚合的方法:(1)排序和(2)哈希。

排序

DBMS 首先根据 GROUP BY 键对元组进行排序。如果所有东西都能放入缓冲池(例如，快速排序)，它可以使用内存排序算法;如果数据的大小超过内存，它可以使用外部归并排序算法。然后，DBMS 对排序后的数据执行顺序扫描，以计算聚合。运算符的输出将根据键进行排序。

在执行排序聚合时，重要的是对查询操作进行排序，以最大化效率。例如，如果查询需要过滤器，最好先执行过滤器，然后对过滤后的数据进行排序，以减少需要排序的数据量。

哈希

在计算聚合时，哈希在计算上比排序更便宜。DBMS 在扫描表时填充一个临时哈希表。对于每条记录，检查哈希表中是否已经有一个条目，并执行适当的修改。如果哈希表的大小太大，内存无法容纳，那么 DBMS 必须将其溢出到磁盘。实现这一过程分为两个阶段:

- **阶段#1 -分区:**使用哈希函数 h1 根据目标哈希键将元组分割为磁盘上的分区。这将把所有匹配的元组放入同一个分区。DBMS 通过输出缓冲区将分区溢出到磁盘。
- **阶段#2 -ReHash:**对于磁盘上的每个分区，将其页面读取到内存中，并基于第二个哈希函数 h2(其中 h1≠h2)构建内存中的哈希表。然后遍历这个哈希表的每个桶，将匹配的元组聚集在一起，计算聚合。这是假设每个分区都在内存中。

在 ReHash 阶段，DBMS 可以存储形式为(GroupByKey→RunningValue)的对来进行计算

聚合。RunningValue 的内容取决于聚合功能。向哈希表中插入一个新的元组:

- 如果找到匹配的 GroupByKey，则适当更新 RunningValue。
- 否则插入新的(GroupByKey→RunningValue)对。