

第9章 关系查询处理和查询优化

9.1 关系数据库系统的查询处理

9.2 关系数据库系统的查询优化

9.3 代数优化

9.4 物理优化

9.4 物理优化

- 代数优化改变查询语句中操作的次序和组合，不涉及底层的存取路径
- 对于一个查询语句有许多存取方案，它们的执行效率不同，仅仅进行代数优化是不够的
- 物理优化就是要选择高效合理的操作算法或存取路径，求得优化的查询计划

• 物理优化方法

• 基于规则的启发式优化

启发式规则是指那些在大多数情况下都适用，但不是在每种情况下都是适用的规则。

• 基于代价估算的优化

优化器估算不同执行策略的代价，并选出具有最小代价的执行计划

- 两者结合的优化方法：

常常先使用启发式规则，选取若干较优的候选方案，减少代价估算的工作量

然后分别计算这些候选方案的执行代价，较快地选出最终的优化方案

9.4 物理优化

9.4.1 基于启发式规则的存取路径选择优化

9.4.2 基于代价的优化

9.4.1 基于启发式规则的存取路径选择优化

1.选择操作的启发式规则

2.连接操作的启发式规则

1.选择操作的启发式规则

- 对于小关系，使用全表顺序扫描，即使选择列上有索引
- 对于大关系，启发式规则有：

(1) 对于选择条件是“主码 = 值”的查询

查询结果最多是一个元组，可以选择主码索引

一般的关系数据库管理系统会自动建立主码索引

(2) 对于选择条件是“非主属性 = 值”的查询，并且选择列上有索引

- 要估算查询结果的元组数目
- 如果比例较小(<10%)可以使用索引扫描方法
- 否则还是使用全表顺序扫描

(3) 对于选择条件是属性上的非等值查询或者范围查询，并且选择列上有索引

- 要估算查询结果的元组数目

如果比例较小($<10\%$)可以使用索引扫描方法

否则还是使用全表顺序扫描

(4) 对于用AND连接的合取选择条件

- 如果有涉及这些属性的组合索引

优先采用组合索引扫描方法

- 如果某些属性上有一般的索引，可以用索引扫描方法

通过分别查找满足每个条件的指针，求指针的交集

通过索引查找满足部分条件的元组，然后在扫描这些元组时判断是否满足剩余条件

- 其他情况：使用全表顺序扫描

(5) 对于用OR连接的析取选择条件，一般使用全表顺序扫描

2.连接操作的启发式规则

(1) 如果2个表都已经按照连接属性排序

- 选用排序-合并算法

(2) 如果一个表在连接属性上有索引

- 选用索引连接算法

(3) 如果上面2个规则都不适用， 其中一个表较小

- 选用Hash join算法

(4) 可以选用嵌套循环方法，并选择其中较小的表，确切地讲是占用的块数(b)较少的表，作为外表(外循环的表)。

理由：

- 设连接表R与S分别占用的块数为 B_r 与 B_s
- 连接操作使用的内存缓冲区块数为 K
- 分配 $K-1$ 块给外表
- 如果R为外表，则嵌套循环法存取的块数为 $B_r + B_r B_s / (K-1)$
- 显然应该选块数小的表作为外表

9.4 物理优化

9.4.1 基于启发式规则的存取路径选择优化

9.4.2 基于代价的优化

9.4.2 基于代价的优化

- 启发式规则优化是定性的选择，适合解释执行的系统
 - 解释执行的系统，优化开销包含在查询总开销之中
- 编译执行的系统中查询优化和查询执行是分开的
 - 可以采用精细复杂一些的基于代价的优化方法



1.统计信息

2.代价估算示例

3.优化方法

1.统计信息

- 基于代价的优化方法要计算查询的各种不同执行方案的执行代价，它与数据库的状态密切相关
- 优化器需要的统计信息
 - (1) 对每个基本表
 - 该表的元组总数(N)
 - 元组长度(l)
 - 占用的块数(B)
 - 占用的溢出块数(BO)

(2) 对基表的每个列

- 该列不同值的个数(m)
- 列最大值
- 最小值
- 列上是否已经建立了索引
- 哪种索引(B+树索引、Hash索引、聚集索引)
- 可以计算选择率(f)
 - ✓ 如果不同值的分布是均匀的, $f = 1/m$
 - ✓ 如果不同值的分布不均匀, 则要计算每个值的选择率, $f = \text{具有该值的元组数}/N$

(3) 对索引

- 索引的层数(L)
- 不同索引值的个数
- 索引的选择基数S(有S个元组具有某个索引值)
- 索引的叶结点数(Y)

2.代价估算示例

(1) 全表扫描算法的代价估算公式

- 如果基本表大小为B块，全表扫描算法的代价 $\text{cost} = B$
- 如果选择条件是“码 = 值”，那么平均搜索代价 $\text{cost} = B/2$

(2) 索引扫描算法的代价估算公式

- 如果选择条件是“码 = 值”

- 则采用该表的主索引

- 若为B+树，层数为L，需要存取B+树中从根结点到叶结点L块，再加上基本表中该元组所在的那一块，所以 $\text{cost} = L + 1$

- 如果选择条件涉及非码属性

- 若为B+树索引，选择条件是相等比较，S是索引的选择基数(有S个元组满足条件)

- 满足条件的元组可能会保存在不同的块上，所以(最坏的情况) $\text{cost} = L + S$

- 如果比较条件是 $>$, $> =$, $<$, $< =$ 操作

- 假设有一半的元组满足条件

- 就要存取一半的叶结点

- 通过索引访问一半的表存储块

- $\text{cost} = L + Y/2 + B/2$

- 如果可以获得更准确的选择基数，可以进一步修正Y/2与B/2

(3) 嵌套循环连接算法的代价估算公式

- 嵌套循环连接算法的代价

$$\text{cost} = B_r + B_r B_s / (K - 1)$$

- 如果要把连接结果写回磁盘

$$\text{cost} = B_r + B_r B_s / (K - 1) + (F_{rs} * N_r * N_s) / M_{rs}$$

- 其中 F_{rs} 为连接选择性(join selectivity), 表示连接结果元组数的比例
- M_{rs} 是存放连接结果的块因子, 表示每块中可以存放的结果元组数目

(4) 排序-合并连接算法的代价估算公式

- 如果连接表已经按照连接属性排好序，则

$$\text{cost} = B_r + B_s + (F_{rs} * N_r * N_s) / M_{rs}$$

- 如果必须对文件排序

- 还需要在代价函数中加上排序的代价
- 对于包含B个块的文件排序的代价大约是

$$(2 * B) + (2 * B * \log_2 B)$$