第12章作业

12.3 设关系 $r_1(A, B, C)$，$r_2(C, D, E)$有如下特性：r_1 有 20 000 个元组，r_2 有 45 000 个元组，一块中可容纳 25 个 r_1 元组或 30 个 r_2 元组。估计使用以下连接策略计算 $r_1 \bowtie r_2$ 需要几次块传输和磁盘搜索：

a. 嵌套循环连接

b. 块嵌套循环连接

c. 归并连接

d. 散列连接

只做(a)和(b)

(a) r_1外层关系、r_2内层关系(最坏)：

块传输：$n_1 \times b_2 + b_1 = 20000 \times \frac{45000}{30} + \frac{20000}{25} = 300000800$

磁盘搜索：$n_1 + b_1 = 20000 + \frac{20000}{25} = 20800$

r_1内层关系、r_2外层关系(最坏)：

块传输：$n_2 \times b_1 + b_2 = 45000 \times \frac{20000}{25} + \frac{45000}{30} = 36001500$

磁盘搜索：$n_2 + b_2 = 45000 + \frac{45000}{30} = 46500$

最好情况：

块传输：$b_1 + b_2 = \frac{20000}{25} + \frac{45000}{30} = 2300$

磁盘搜索：2

(b) r_1外层关系、r_2内层关系(最坏)：

块传输：$b_1 \times b_2 + b_1 = \frac{20000}{25} \times \frac{45000}{30} + \frac{20000}{25} = 1200800$

磁盘搜索：$2b_1 = 2 \times \frac{20000}{25} = 1600$

r_1内层关系、r_2外层关系(最坏)：

块传输：$b_2 \times b_1 + b_2 = \frac{45000}{30} \times \frac{20000}{25} + \frac{45000}{30} = 1201500$

磁盘搜索：$2b_2 = 2 \times \frac{45000}{30} = 3000$

最好情况：

块传输：$b_1 + b_2 = \frac{20000}{25} + \frac{45000}{30} = 2300$

磁盘搜索：2

select *T.branch_name*
from *branch T*, *branch S*
where *T.assets* > *S.assets* **and** *S.branch_city* = "Brooklyn"

Write an efficient relational-algebra expression that is equivalent to this query. Justify your choice.

Answer:

Query:

$$\Pi_{T.branch\_name}((\Pi_{branch\_name,\, assets}(\rho_T(branch))) \bowtie_{T.assets\, >\, S.assets} (\Pi_{assets}\ (\sigma_{(branch\_city\, =\, \text{'Brooklyn'})}(\rho_S(branch)))))$$

This expression performs the theta join on the smallest amount of data possible. It does this by restricting the right hand side operand of the join to only those branches in Brooklyn, and also eliminating the unneeded attributes from both the operands.

12.3 Let relations $r_1(A, B, C)$ and $r_2(C, D, E)$ have the following properties: r_1 has 20,000 tuples, r_2 has 45,000 tuples, 25 tuples of r_1 fit on one block, and 30 tuples of r_2 fit on one block. Estimate the number of block transfers and seeks required, using each of the following join strategies for $r_1 \bowtie r_2$:

a. Nested-loop join.

b. Block nested-loop join.

c. Merge join.

d. Hash join.

Answer:

r_1 needs 800 blocks, and r_2 needs 1500 blocks. Let us assume M pages of memory. If $M > 800$, the join can easily be done in 1500 + 800 disk accesses, using even plain nested-loop join. So we consider only the case where $M \leq 800$ pages.

a. Nested-loop join:
Using r_1 as the outer relation we need $20000 * 1500 + 800 = 30,000,800$ disk accesses, if r_2 is the outer relation we need $45000 * 800 + 1500 = 36,001,500$ disk accesses.

b. Block nested-loop join:
If r_1 is the outer relation, we need $\lceil \frac{800}{M-1} \rceil * 1500 + 800$ disk accesses, if r_2 is the outer relation we need $\lceil \frac{1500}{M-1} \rceil * 800 + 1500$ disk accesses.

c. Merge-join:
Assuming that r_1 and r_2 are not initially sorted on the join key, the total sorting cost inclusive of the output is $B_s = 1500(2\lceil log_{M-1}(1500/M) \rceil +$