ABSTRACT

To find the optimal plan to process the query by evaluating query processing costs for all possible execution plans for each Q1 and RQ1 and comparing the costs to choose the minimum cost plan at the end.

SPECIFICATION

We have written a java program which calculates the query processing cost and then selects the best plan, on basis of join and order.

The following are the joins that are considered as per the given requirements:

1. Tuple Nested Loop Join: TNL
2. Page Nested Loop Join: PNL
3. Block Nested Loop Join: with Buffer Memory: BNJM
4. Sort Merge Join with buffer memory: SMJM
5. Hash Join with Buffer: HJM
6. Hash Join with less Buffer: HJL
7. Block Nested Loop Join with less Buffer Memory: BNJL
8. Sort Merge Join with less buffer memory: SMJL

Two queries for calculation as follow,

A possible Query Execution Steps of Q1:

```
1 Join t1 t2                    : any join methods for (temp1 <= t1 join t2)
2 Join temp1 t3                 :
        For each tuple of temp1: only for TNJ (because of correlation) with
                                 two selectivities 10 % * 20 % for the result
                                 temp2 ← temp1 join t3
                                 on the predicates T2.x2 = T3.x3 : 10% and
                                 T1.x1 = T3.x3 : 20 %

        3 Project temp2          : temp3 ← temp2
        4.Aggregate without Group By

4 GroupBy temp3 with Aggregation
```

A possible Query Execution Steps of RQ1:

A possible Execution Steps for that:
1 Join t1 t3 : any join methods for (temp1 ← t1 join t3) with selectivity 20 % on T1.x1 = T3.x3 for the result temp0 ← t1 join t3

2.Group By temp0 on T1.Rowid with Aggregate →Result table called Temp1

3 Join t1 Temp1 : any join methods for (temp2 ← t1 join Temp1) with selectivity 15 % on T1.Rowid = temp1.rowid for the result temp2

4. Join t2 temp2 : any join methods for (temp3 <= t2 join temp2) with selectivity 10 % on T2.x2 = temp2.x3 for the result temp3

5 Project temp3 : temp3 <= temp2
6. GROUP BY temp3. X1 with Aggregate

PROJECT OVERVIEW

We are taking the input from the user regarding the tuple size and buffer size and query and then calculation the join cost on basis of entered parameter.

We created two classes,
1. OptimiserQueryConfigureBox
2. SQLProcess

DESIGN AND IMPLEMENTATION

The program reads the input from text area and performs the join operation calculation. This join operation calculation includes several processes. It calls all mentioned above 8 types of join one by one and then saves the least costing join and its values. Also, it considers the order of join saves the result.

Consider different join cost calculation:

Tuple Nested Loop Join: TNL
TNL calculation Cost: M (to scan R) + (pR * M) times * N (to scan S)

Page Nested Loop Join: PNL
Page-oriented Nested Loops join Cost: M (to scan R) + M times *N

Sort Merge Join with buffer memory
SJM (Consider bufferMemory = 50 and less memory 30)
SMJ cost = 2M (1+ [logB-1 [M/B]]) + 2N(1+ [logB-1 [N/B]]) + M + N

Hash Join with Buffer memory
HJM (Consider buffer memory = 50 and less memory 30)
In partitioning phase, read and write operations; each has to read 2(M+N).
In matching phase, we need match only once; M+N I/Os.
Total Cost: 3(M + N) $int\ cost$ =(3∗(m+n))
But for specified buffer memory = 50
Total Cost: 2 * (M+N) * (1+logB-1(M+N)/B-1) + M + N

Aggregation of results
Group By with aggregation is given calculated as follows:
Group by (with Aggregation on the fly at the last scan) is Sorting cost

Both Sorting base algorithms and Hash based algorithms for Group By costs 3(M + N)
If last step is Sort Merge Join,
then Group by Cost is 2(M+N)

Considering the above operations for the calculating the joining cost. We will first make sure that the optimal join and order is picked, then we will calculate the query processing cost which is given by
Query Processing Cost = Disk I/O Cost
= # of Disk I/O * Disk Access Time
= # of Disk I/O * (8 ms + 4 ms)
= Total # of Disk Block access needed * 12ms
The result will be in milliseconds and we will display the result in hours: minutes: seconds format.

INPUT

Q1.txt

```
Join t1 t2
Join temp1 t3
Project temp2
GroupBy temp3
```

RQ1.txt

```
Join t1 t3
GroupBy temp0
Join t1 Temp1
Join t2 temp2
Project temp3
GROUPBY temp3
```

OUTPUT