Query processing cost formulae

Legend

Symbol	Description
NKeys(Col)	The number of distinct values of column Col
High(Col)	The highest value of column Col
Low(Col)	The lowest value of column Col
NTuples(R)	The number of tuples of relation R
NPages(R)	The number of pages of relation R
NPages(I)	The number of pages of index I
Height(I)	The height of index I
Π **RF$_i$**	The product of all reduction factors
Π **NTuples(R$_i$)**	The product of the numbers of tuples of all relations taking part in a join
ceil(1+ log$_{B-1}$ ceil(NPages(R)/B))	The number of passes for sorting

1. **Reduction factor (Selectivity)**
   a. Col = value

      RF = 1/NKeys(Col)

   b. Col > value

      RF = (High(Col) – value) / (High(Col) – Low(Col))

   c. Col < value

      RF = (val – Low(Col)) / (High(Col) – Low(Col))

   d. Col_A = Col_B (for joins)

      RF = 1/ (Max (NKeys(Col_A), NKeys(Col_B)))

   e. In no information about NKeys, use a “magic number” 1/10

      RF = 1/10

2. **Result size calculations**
   a. Single table

      Result_size = NTuples(R) * Π RF$_i$

   b. Joins

      Result_size = Π NTuples(R$_i$) * Π RF$_i$

3. **Indexing**
    a. B+-tree index
        i. Just a single tuple (selection over a primary key)

        Cost = Height(I) + 1

        ii. Clustered index (multiple tuples)

        Cost = (NPages(I)+NPages(R)) * Π RF_i

        iii. Unclustered (multiple tuples)

        Cost = (NPages(I)+NTuples(R)) * Π RF_i

    b. Hash Index
        i. Just a single tuple (selection over a primary key)

        Cost = 1.2 + 1 = 2.2

        ii. Clustered index (multiple tuples)

        Cost = (NPages (R)) * Π RF_i * 2.2

        iii. Unclustered index (multiple tuples)

        Cost = (NTuples(R)) * Π RF_i * 2.2

4. **Sequential Scan (i.e. Heap Scan)**

    Cost = NPages(R)

5. **Joins (between relations R and S, R = outer, S = inner)**
   a. NLJ
      i. Tuple-oriented NLJ

         Cost = NPages(R) + NTuples(R) * NPages(S)

      ii. Page-oriented NLJ

         Cost = NPages(R) + NPages (R) * NPages(S)

      iii. Block-oriented NJL (for block_size B)

         Cost = NPages(R) + ceil(NPages (R)/(B-2)) * NPages(S)

      iv. Index NLJ

         Cost = NPages(R) + NTuples(R) * cost of a single tuple of S

   b. Hash Join

      Cost = NPages(R) + NPages(S) + 2*(NPages(R) + NPages(S))

      Note: 2*(NPages(R) + NPages(S)) is for partitioning and is optional if one relation fits entirely in memory. In that case only hashing will happen.

   c. Sort-Merge Join (for block_size B)

      $Cost_{SMJ}$ = NPages(R) + NPages(S) +

      2* NPages(R)* ceil(1+ $\log_{B-1}$ ceil(NPages(R)/B)) +

      2* NPages(S)* ceil(1+ $\log_{B-1}$ ceil(NPages(S)/B))

      c.1 Improvement: do not sort completely, but merge sorted runs directly (both formulae will be accepted!)

      $Cost_{SMJ\_improved}$ = $Cost_{SMJ}$ – 2*(NPages(R)) – 2*(NPages(S))