



Database System Concept (CSE 3103)

Lecture 07-Day 02

Nazmus Sakib, Assistant Professor, Dept. of CSE, AUST

Basic Steps in Query Processing : Optimization

- A relational algebra expression may have many equivalent expressions
 - E.g., $\sigma_{salary < 75000}(\Pi_{salary}(instructor))$ is equivalent to $\Pi_{salary}(\sigma_{salary < 75000}(instructor))$
- Each relational algebra operation can be evaluated using one of several different algorithms
 - Correspondingly, a relational-algebra expression can be evaluated in many ways.
- Annotated expression specifying detailed evaluation strategy is called an **evaluation-plan**.
 - E.g., can use an index on *salary* to find instructors with salary < 75000,
 - or can perform complete relation scan and discard instructors with salary ≥ 75000

Basic Steps: Optimization (Cont.)

- **Query Optimization:** Amongst all equivalent evaluation plans choose the one with lowest cost.
 - Cost is estimated using statistical information from the database catalog
 - e.g. number of tuples in each relation, size of tuples, etc.
- In this chapter we study
 - How to measure query costs
 - Algorithms for evaluating relational algebra operations
 - How to combine algorithms for individual operations in order to evaluate a complete expression
- In Chapter 14
 - We study how to optimize queries, that is, how to find an evaluation plan with lowest estimated cost

Measures of Query Cost

- Cost is generally measured as total elapsed time for answering query
 - Many factors contribute to time cost
 - *disk accesses, CPU, or even network communication*
- Typically disk access is the predominant cost, and is also relatively easy to estimate. Measured by taking into account
 - Number of seeks * average-seek-cost
 - Number of blocks read * average-block-read-cost
 - Number of blocks written * average-block-write-cost
 - Cost to write a block is greater than cost to read a block
 - data is read back after being written to ensure that the write was successful

Measures of Query Cost (Cont.)

- For simplicity we just use the **number of block transfers** *from disk* and the **number of seeks** as the cost measures
 - t_T – time to transfer one block
 - t_S – time for one seek
 - Cost for b block transfers plus S seeks
$$b * t_T + S * t_S$$
- We ignore CPU costs for simplicity
 - Real systems do take CPU cost into account
- We do not include cost to writing output to disk in our cost formulae

Selections Using Indices

- **A4 (secondary index, equality on nonkey).**
 - Retrieve a single record if the search-key is a candidate key
 - $Cost = (h_i + 1) * (t_T + t_S)$
 - Retrieve multiple records if search-key is not a candidate key
 - each of n matching records may be on a different block
 - $Cost = (h_i + n) * (t_T + t_S)$
 - Can be very expensive!

Sorting

- We may build an index on the relation, and then use the index to read the relation in sorted order. May lead to one disk block access for each tuple.
- For relations that fit in memory, techniques like quicksort can be used. For relations that don't fit in memory, **external sort-merge** is a good choice.