DataBase Systems

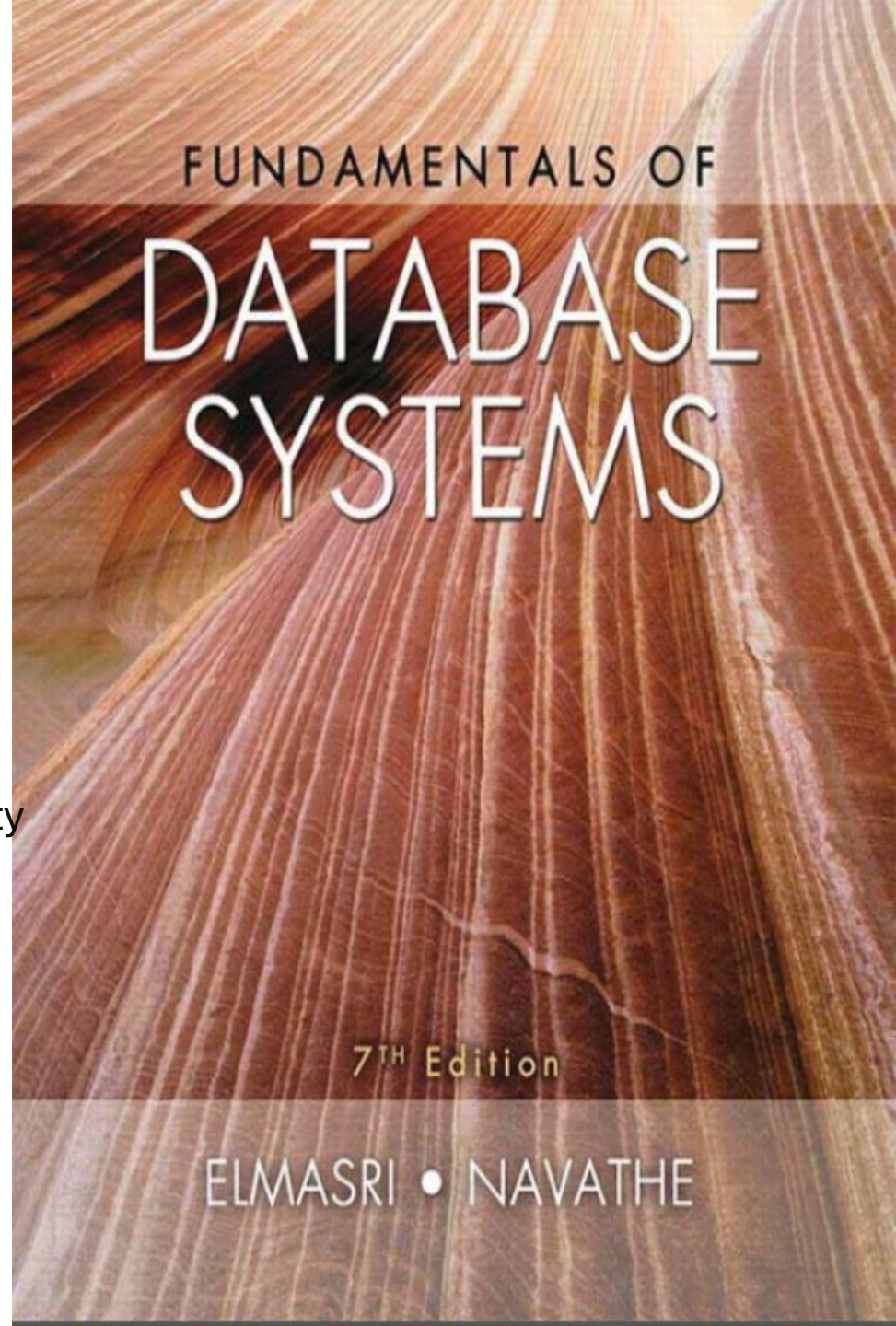
Chapter 19

Query Optimization

Dr. Ahmed Anter

Associate Professor – Egypt-Japan University of Science and Technology (E-JUST)

Spring, 2025



Introduction

- Query optimization
  - Conducted by a query optimizer in a DBMS
  - Goal: select best available strategy for executing query
    - Based on information available
- Most RDBMSs use a tree as the internal representation of a query

19.1 Query Trees and Heuristics for Query Optimization

- Step 1: scanner and parser generate initial query representation
- Step 2: representation is optimized according to heuristic rules
- Step 3: query execution plan is developed
  - Execute groups of operations based on access paths available and files involved

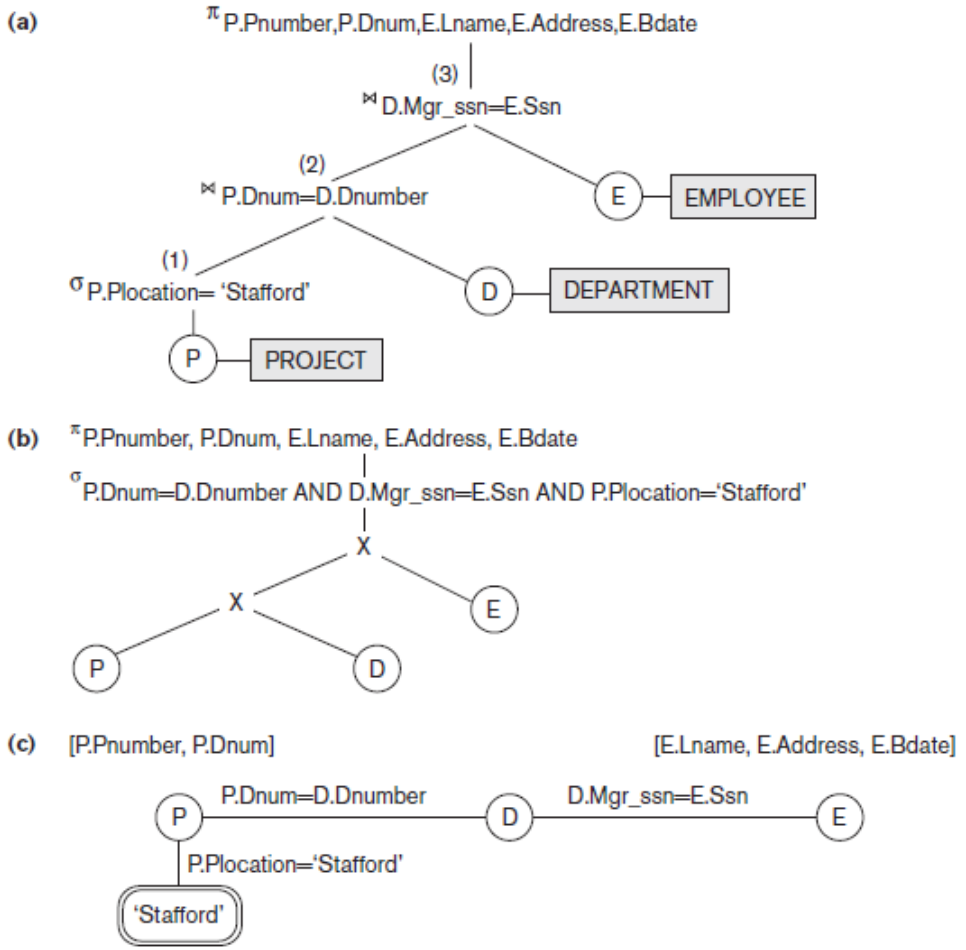
Query Trees and Heuristics for Query Optimization (cont'd.)

- Example heuristic rule
  - Apply SELECT and PROJECT before JOIN
    - Reduces size of files to be joined
- Query tree
  - Represents relational algebra expression
- Query graph
  - Represents relational calculus expression
- Example for Q2 on next slide

```
Q2:  SELECT   P.Pnumber, P.Dnum, E.Lname, E.Address, E.Bdate
     FROM     PROJECT P, DEPARTMENT D, EMPLOYEE E
     WHERE    P.Dnum=D.Dnumber AND D.Mgr_ssn=E.Ssn AND
              P.Plocation= 'Stafford';
```

Query Trees and Query Graph Corresponding to Q2

Figure 19.1 Two query trees for the query Q2. (a) Query tree corresponding to the relational algebra expression for Q2. (b) Initial (canonical) query tree for SQL query Q2. (c) Query graph for Q2.



Query Trees and Heuristics for Query Optimization (cont'd.)

- Query tree represents a specific order of operations for executing a query
  - Preferred to query graph for this reason
- Query graph
  - Relation nodes displayed as single circles
  - Constants represented by constant nodes
    - Double circles or ovals
  - Selection or join conditions represented as edges
  - Attributes to be retrieved displayed in square brackets

Heuristic Optimization of Query Trees

- Many different query trees can be used to represent the query and get the same results
- Figure 19.1b shows initial tree for Q2
  - Very inefficient - will never be executed
  - Optimizer will transform into equivalent final query tree

Query Transformation Example

```
Q:  SELECT  E.Lname
    FROM    EMPLOYEE E, WORKS_ON W, PROJECT P
    WHERE   P.Pname='Aquarius' AND P.Pnumber=W.Pno AND E.Essn=W.Ssn
            AND E.Bdate > '1957-12-31';
```

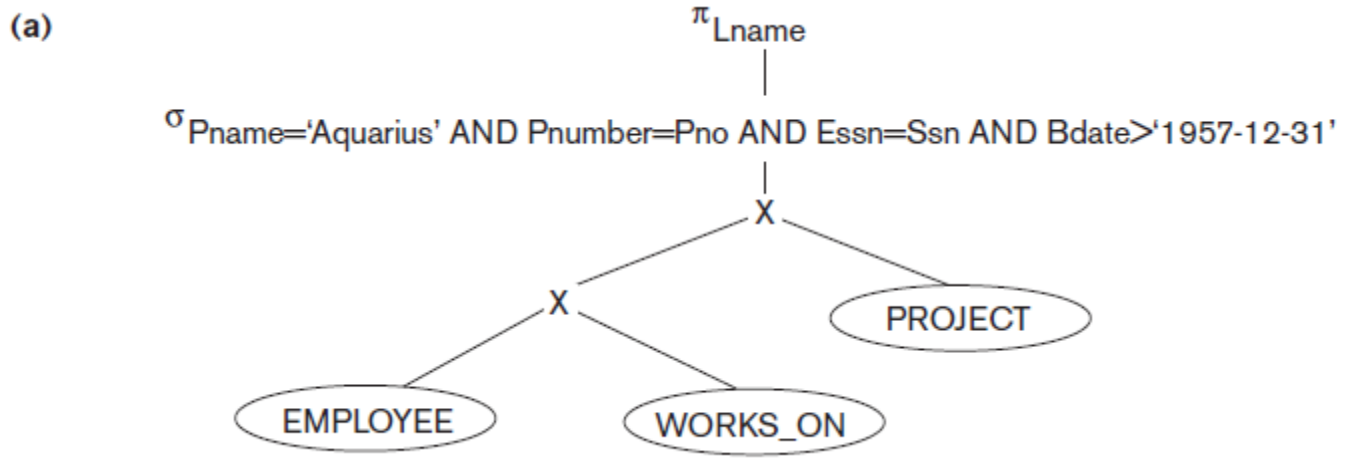


Figure 19.2 Steps in converting a query tree during heuristic optimization. (a) Initial (canonical) query tree for SQL query Q.

Query Transformation Example (cont'd.)

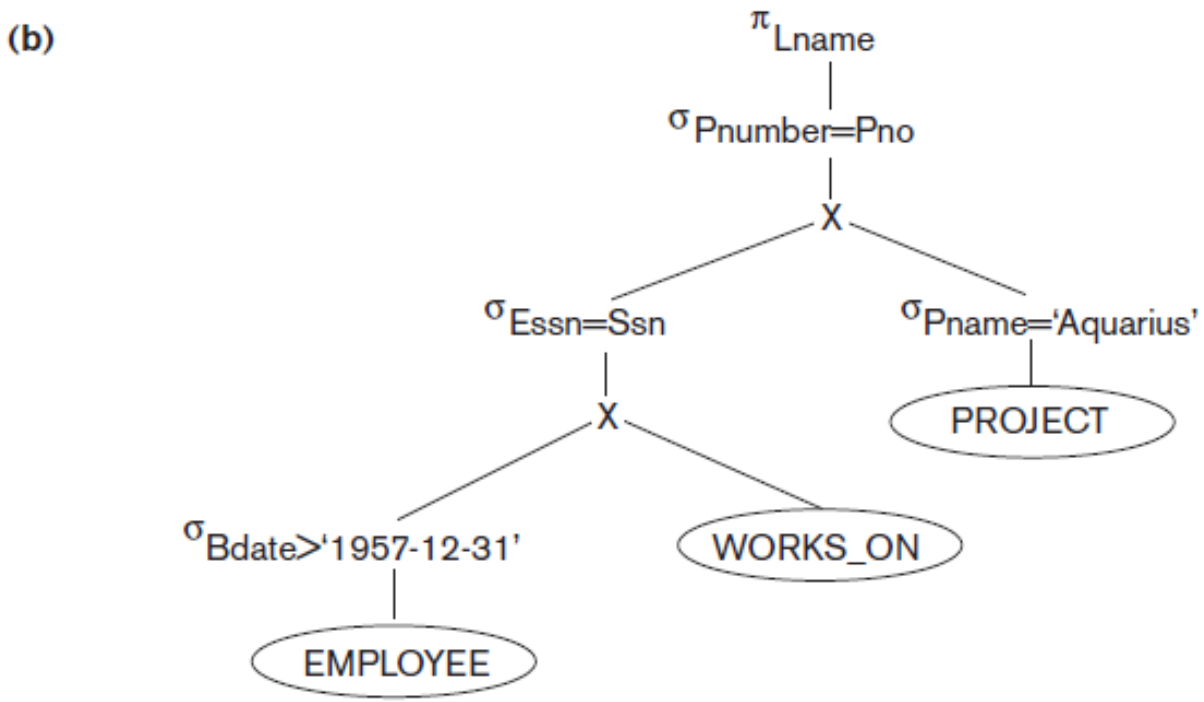


Figure 19.2 Steps in converting a query tree during heuristic optimization (b) Moving SELECT operations down the query tree.

Query Transformation Example (cont'd.)

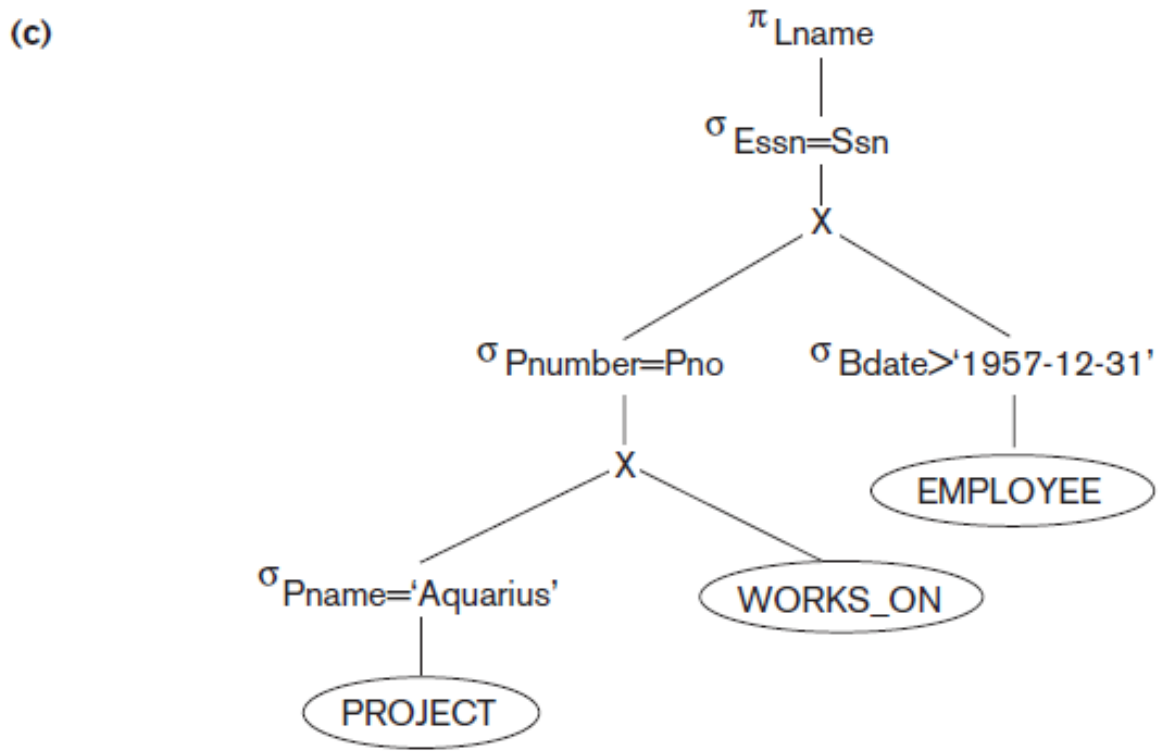


Figure 19.2 Steps in converting a query tree during heuristic optimization (c) Applying the more restrictive SELECT operation first.

Query Transformation Example (cont'd.)

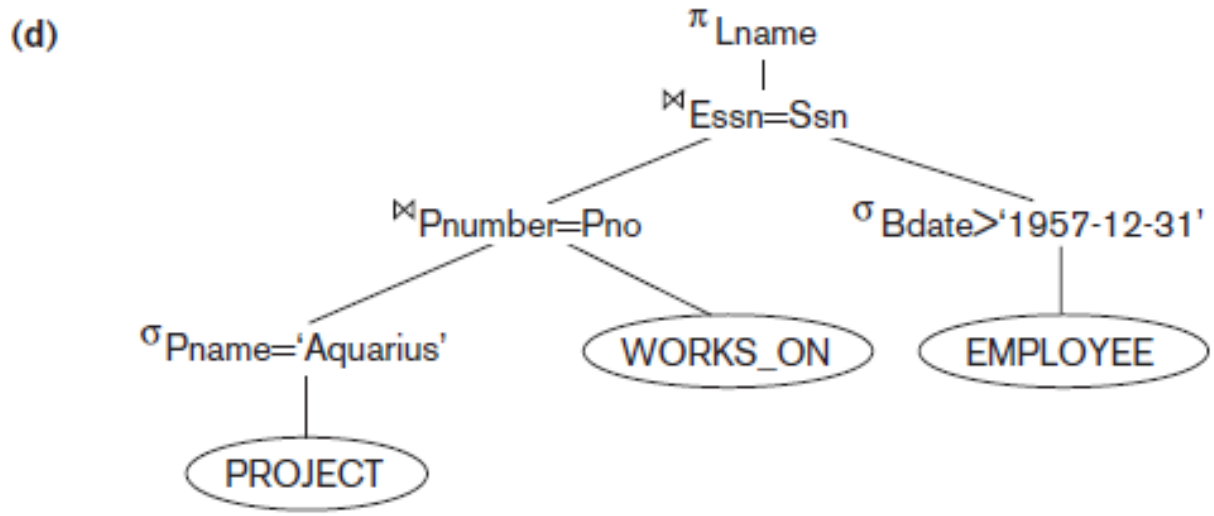


Figure 19.2 Steps in converting a query tree during heuristic optimization
(d) Replacing CARTESIAN PRODUCT and SELECT with JOIN operations.

Query Transformation Example (cont'd.)

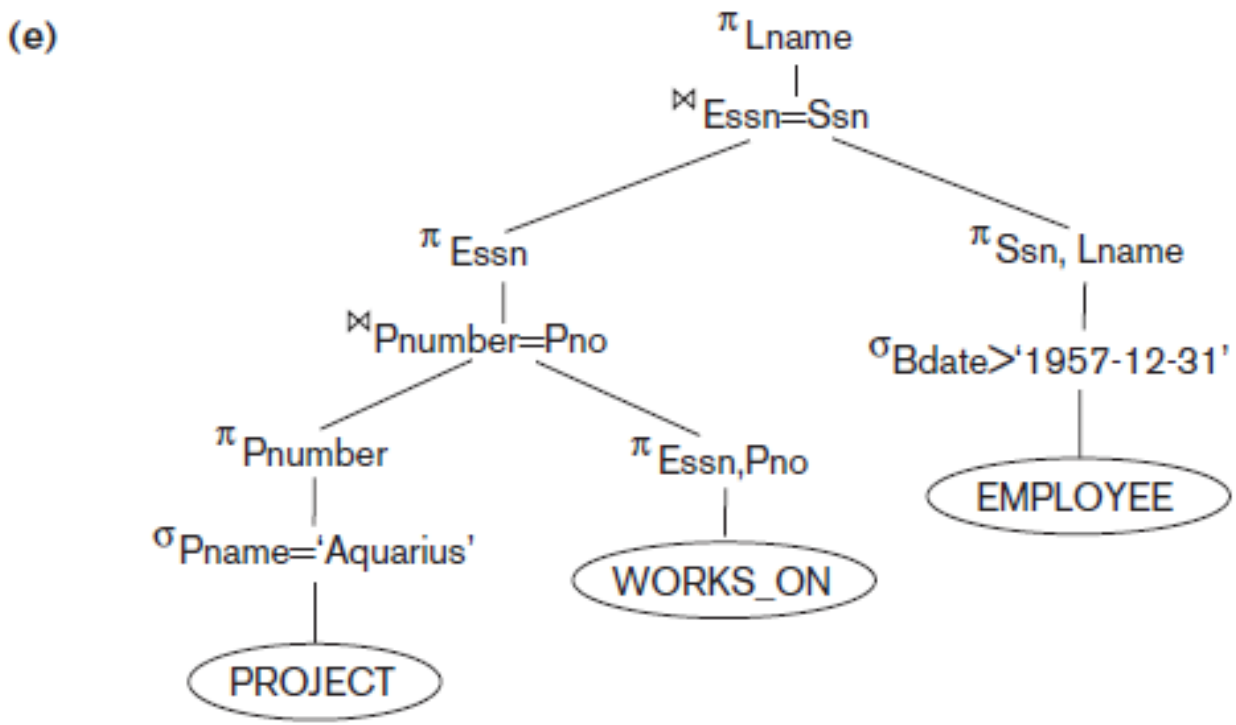


Figure 19.2 Steps in converting a query tree during heuristic optimization (e) Moving PROJECT operations down the query tree.

General Transformation Rules for Rational Algebra Equations

- Some transformation rules useful in query optimization
  - Cascade of σ
    - Conjunctive selection condition can be broken up into a cascade (sequence) of individual σ operations
  - Commutativity of σ
  - Cascade of π
    - In a sequence of π operations, all but the last one can be ignored
  - Commuting σ with π

Summary of Heuristics for Algebraic Optimization

- Apply first the operations that reduce the size of intermediate results
  - Perform SELECT and PROJECT operations as early as possible to reduce the number of tuples and attributes
  - The SELECT and JOIN operations that are most restrictive should be executed before other similar operations

19.2 Choice of Query Execution Plans

- Materialized evaluation
  - Result of an operation stored as temporary relation
- Pipelined evaluation
  - Operation results forwarded directly to the next operation in the query sequence

Nested Subquery Optimization

- Unnesting
  - Process of removing the nested query and converting the inner and outer query into one block
- Queries involving a nested subquery connected by IN or ANY connector can always be converted into a single block query
- Alternate technique
  - Creating temporary result tables from subqueries and using them in joins

Subquery (View) Merging Transformation

- Inline view
  - FROM clause subquery
- View merging operation
  - Merges the tables in the view with the tables from the outer query block
  - Views containing select-project-join operations are considered simple views
    - Can always be subjected to this type of view-merging

Subquery (View) Merging Transformation (cont'd.)

- Group-By view-merging
  - Delaying the Group By operation after performing joins may reduce the data subjected to grouping in case the joins have low join selectivity
  - Alternately, performing Group By early may reduce the amount of data subjected to subsequent joins
  - Optimizer determines whether to merge GROUP-BY views based on estimated costs

Materialized Views

- View defined in database as a query
  - Materialized view stores results of that query
    - May be stored temporarily or permanently
- Optimization technique
  - Using materialized views to avoid some of the computation involved in the query
  - Easier to read it when needed than recompute from scratch

Incremental View Maintenance

- Update view incrementally by accounting for changes that occurred since last update
  - Join
  - Selection
  - Projection
  - Intersection
  - Aggregation

19.3 Use of Selectives in Cost-Based Optimization

- Query optimizer estimates and compares costs of query execution using different strategies
  - Chooses lowest cost estimate strategy
  - Process suited to compiled queries
- Interpreted queries
  - Entire process occurs at runtime
  - Cost estimate may slow down response time

Use of Selectives in Cost-Based Optimization (cont'd.)

- Cost-based query optimization approach
  - For a given query subexpression, multiple equivalence rules may apply
  - Quantitative measure for evaluating alternatives
    - Cost metric includes space and time requirements
  - Design appropriate search strategies by keeping cheapest alternatives and pruning costlier alternatives
  - Scope of query optimization is a query block
    - Global query optimization involves multiple query blocks

Use of Selectives in Cost-Based Optimization (cont'd.)

- Cost components for query execution
  - Access cost to secondary storage
  - Disk storage cost
  - Computation cost
  - Memory usage cost
  - Communication cost

Catalog Information Used in Cost Functions

- Information stored in DBMS catalog and used by optimizer
  - File size
  - Organization
  - Number of levels of each multilevel index
  - Number of distinct values of an attribute
  - Attribute selectivity
    - Allows calculation of selection cardinality
      - Average number of records that satisfy equality selection condition on that attribute

Histograms

- Tables or data structures that record information about the distribution of data
- RDBMS stores histograms for most important attributes

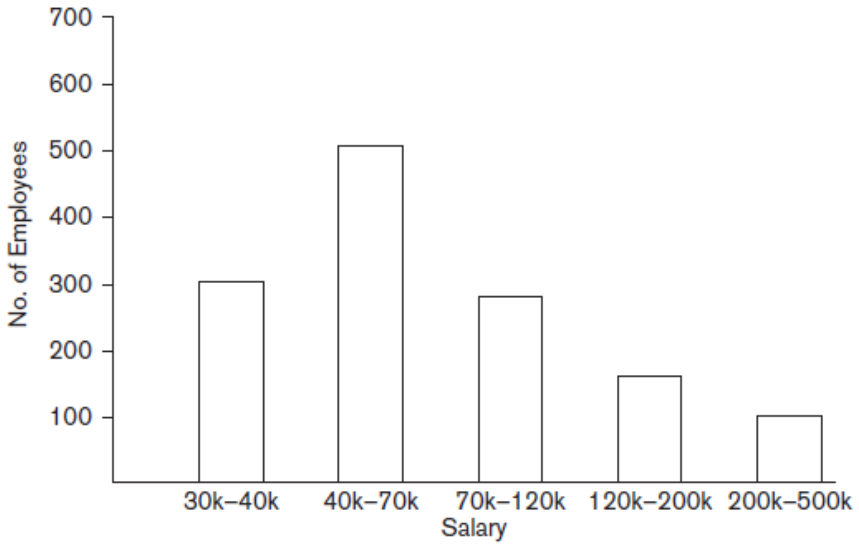


Figure 19.4 Histogram of salary in the relation EMPLOYEE

19.11 Summary

- Query trees
- Heuristic approaches used to improve efficiency of query execution
- Reorganization of query trees
- Pipelining and materialized evaluation
- Cost-based optimization approach
- Oracle query optimizer
- Semantic query optimization