

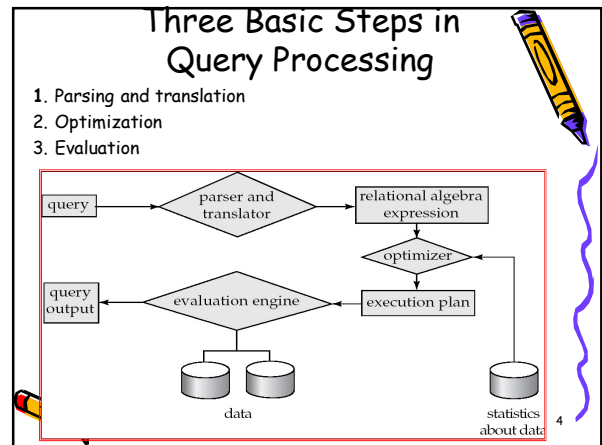



# Query Processing and Optimisation


1

- ## Query Processing
- What is Query Processing
  - Measures of Query Cost
  - Selection Operation
  - Sorting
- 
- 2

- ## What is Query Processing?
- Query processing: Activities involved in extracting data from a database.
    - Translation of queries in high-level DB languages into expressions that can be used at physical level of file system.
    - Includes query optimization and query evaluation.
  - Three basic steps:
    1. Parsing and Translation
    2. Optimization
    3. Evaluation
- 
- 3



- ## Parsing and translation
- Translate the query into its internal form.
    - This is then translated into relational algebra.
  - Parser checks syntax, verifies relations.
- 
- 5

- ## Parsing and translation
- A relational algebra expression may have many equivalent expressions
    - E.g.,  $\sigma_{balance < 2500}(\Pi_{balance}(account))$  is equivalent to
- $$\Pi_{balance}(\sigma_{balance < 2500}(account))$$
- 
- 6

### Parsing and translation (cont.)

- Each relational algebra operation can be evaluated using one of several different algorithms
- Correspondingly, a relational-algebra expression can be evaluated in many ways.
- Evaluation-plan:** Annotated expression specifying detailed evaluation strategy.
  - e.g., can use an index on *balance* to find accounts with *balance* < 2500,
  - or can perform complete relation scan and discard accounts with *balance* ≥ 2500

7

### Query Optimization

- Alternative ways of evaluating a given query
  - Equivalent expressions
  - Different algorithms for each operation

8

### Query Optimization

- An **evaluation plan** defines exactly what algorithm is used for each operation, and how the execution of the operations is coordinated.

9

### 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.
  - How to measure query costs
  - How to optimize queries, that is, how to find an evaluation plan with lowest estimated cost

10

### Query Optimization

- Estimation of plan cost based on:
  - Statistical information about relations. Examples:
    - number of tuples, number of distinct values for an attribute
  - Statistics estimation for intermediate results
    - to compute cost of complex expressions
  - Cost formulae for algorithms, computed using statistics

11


### Query Optimization

- Cost difference between evaluation plans for a query can be enormous
  - E.g. seconds vs. days in some cases
- Steps in cost-based query optimization
  - Generate logically equivalent expressions using equivalence rules
  - Annotate resultant expressions to get alternative query plans
  - Choose the cheapest plan based on estimated cost

12

### Evaluation


- The query-execution engine takes a query-evaluation plan, executes that plan, and returns the answers to the query.
- Parsed execution plan for previously executed SQL statements is stored in Shared pool (a portion of memory or buffer).
  - If a new SQL statement (query) is exactly the same string as the one in the shared pool, no need to call optimizer and recalculate the execution plan for the SQL statement.



13

### Transformation of Relational Expressions


- Two relational algebra expressions are said to be **equivalent** if the two expressions generate the same set of tuples on every legal database instance
  - Note: order of tuples is irrelevant



14

### Transformation of Relational Expressions


- An **equivalence rule** says that expressions of two forms are equivalent
  - Can replace expression of first form by second, or vice versa



15

### Equivalence Rules


- Conjunctive selection operations can be deconstructed into a sequence of individual selections.
 
$$\sigma_{\theta_1 \wedge \theta_2}(E) = \sigma_{\theta_1}(\sigma_{\theta_2}(E))$$
- Selection operations are commutative.
 
$$\sigma_{\theta_1}(\sigma_{\theta_2}(E)) = \sigma_{\theta_2}(\sigma_{\theta_1}(E))$$



16

### Equivalence Rules (Cont.)


- Only the last in a sequence of projection operations is needed, the others can be omitted.
 
$$\Pi_{L_1}(\Pi_{L_2}(\dots(\Pi_{L_n}(E))\dots)) = \Pi_{L_1}(E)$$
- Selections can be combined with Cartesian products and theta joins.
  - $\sigma_{\theta}(E_1 \times E_2) = E_1 \bowtie_{\theta} E_2$
  - $\sigma_{\theta_1}(E_1 \bowtie_{\theta_2} E_2) = E_1 \bowtie_{\theta_1 \wedge \theta_2} E_2$



17

### Equivalence Rules (Cont.)

- The selection operation distributes over the theta join operation under the following two conditions:
  - When all the attributes in  $\theta_0$  involve only the attributes of one of the expressions ( $E_i$ ) being joined.
 
$$\sigma_{\theta_0}(E_1 \bowtie_{\theta} E_2) = (\sigma_{\theta_0}(E_1)) \bowtie_{\theta} E_2$$
  - When  $\theta_1$  involves only the attributes of  $E_1$  and  $\theta_2$  involves only the attributes of  $E_2$ .
 
$$\sigma_{\theta_1 \wedge \theta_2}(E_1 \bowtie_{\theta} E_2) = (\sigma_{\theta_1}(E_1)) \bowtie_{\theta} (\sigma_{\theta_2}(E_2))$$



18

### Transformation Example: Pushing Selections

- Query: Find the names of all customers who have an account at some branch located in Brooklyn.

$$\Pi_{customer\_name}(\sigma_{branch\_city = "Brooklyn"}(branch \bowtie (account \bowtie depositor)))$$

- Transformation using rule 5a

$$\Pi_{customer\_name}((\sigma_{branch\_city = "Brooklyn"}(branch)) \bowtie (account \bowtie depositor))$$

19

### Transformation Example: Pushing Selections

- Performing the selection as early as possible reduces the size of the relation to be joined.

20

### Cost Estimation

- Cost of each operator computer
  - Need statistics of input relations
    - E.g. number of tuples, sizes of tuples
- Inputs can be results of sub-expressions
  - Need to estimate statistics of expression results
  - To do so, we require additional statistics
    - E.g. number of distinct values for an attribute

21

### Measures of Query Cost

- Cost is generally measured as total elapsed time for answering query
- Factors contribute to time cost
  - Disk accesses
    - How does the index/hashing approach impact?
  - CPU
  - Network communication

22

### Measures of Query Cost

- 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

23


### Measures of Query 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

24

### Selection Operation


- Let start with a select query
- **File scan** - search algorithms that locate and retrieve records that fulfill a selection condition.
- Two ways to accomplish
  - Algorithm A1 → linear search
  - Algorithm A2 → binary search



25

### Selections Using Indices

- **Index scan** - search algorithms that use an index
  - selection condition must be on search-key of index.
- **Algorithm A3** (primary index on candidate key, equality). Retrieve a single record that satisfies the corresponding equality condition




26

### Sorting

Sorting is useful not only to return sorted data to users but also to facilitate join.

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.



27

### Sorting

- 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.

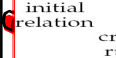


28

### Example: External Sorting Using Sort-Merge

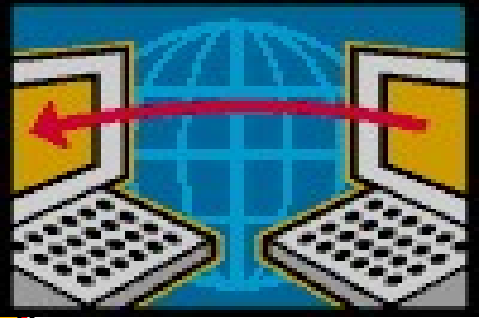

<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>g</td><td>24</td></tr> <tr><td>a</td><td>19</td></tr> <tr><td>d</td><td>31</td></tr> <tr><td>e</td><td>33</td></tr> <tr><td>b</td><td>14</td></tr> <tr><td>e</td><td>16</td></tr> <tr><td>r</td><td>16</td></tr> <tr><td>d</td><td>21</td></tr> <tr><td>m</td><td>3</td></tr> <tr><td>p</td><td>2</td></tr> <tr><td>d</td><td>7</td></tr> <tr><td>a</td><td>14</td></tr> </table> <p style="text-align: center;">initial relation</p>	g	24	a	19	d	31	e	33	b	14	e	16	r	16	d	21	m	3	p	2	d	7	a	14	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>a</td><td>19</td></tr> <tr><td>d</td><td>31</td></tr> <tr><td>g</td><td>24</td></tr> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>b</td><td>14</td></tr> <tr><td>c</td><td>33</td></tr> <tr><td>e</td><td>16</td></tr> </table> <p style="text-align: center;">runs</p>	a	19	d	31	g	24	b	14	c	33	e	16	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>a</td><td>19</td></tr> <tr><td>b</td><td>14</td></tr> <tr><td>c</td><td>33</td></tr> <tr><td>d</td><td>31</td></tr> <tr><td>e</td><td>16</td></tr> <tr><td>g</td><td>24</td></tr> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>a</td><td>14</td></tr> <tr><td>d</td><td>7</td></tr> <tr><td>d</td><td>21</td></tr> <tr><td>m</td><td>3</td></tr> <tr><td>p</td><td>2</td></tr> <tr><td>r</td><td>16</td></tr> </table> <p style="text-align: center;">runs</p>	a	19	b	14	c	33	d	31	e	16	g	24	a	14	d	7	d	21	m	3	p	2	r	16	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>a</td><td>14</td></tr> <tr><td>a</td><td>19</td></tr> <tr><td>b</td><td>14</td></tr> <tr><td>c</td><td>33</td></tr> <tr><td>d</td><td>7</td></tr> <tr><td>d</td><td>21</td></tr> <tr><td>d</td><td>31</td></tr> <tr><td>e</td><td>16</td></tr> <tr><td>g</td><td>24</td></tr> <tr><td>m</td><td>3</td></tr> <tr><td>p</td><td>2</td></tr> <tr><td>r</td><td>16</td></tr> </table> <p style="text-align: center;">sorted output</p>	a	14	a	19	b	14	c	33	d	7	d	21	d	31	e	16	g	24	m	3	p	2	r	16
g	24																																																																																						
a	19																																																																																						
d	31																																																																																						
e	33																																																																																						
b	14																																																																																						
e	16																																																																																						
r	16																																																																																						
d	21																																																																																						
m	3																																																																																						
p	2																																																																																						
d	7																																																																																						
a	14																																																																																						
a	19																																																																																						
d	31																																																																																						
g	24																																																																																						
b	14																																																																																						
c	33																																																																																						
e	16																																																																																						
a	19																																																																																						
b	14																																																																																						
c	33																																																																																						
d	31																																																																																						
e	16																																																																																						
g	24																																																																																						
a	14																																																																																						
d	7																																																																																						
d	21																																																																																						
m	3																																																																																						
p	2																																																																																						
r	16																																																																																						
a	14																																																																																						
a	19																																																																																						
b	14																																																																																						
c	33																																																																																						
d	7																																																																																						
d	21																																																																																						
d	31																																																																																						
e	16																																																																																						
g	24																																																																																						
m	3																																																																																						
p	2																																																																																						
r	16																																																																																						

create runs
merge pass-1
merge pass-2



29

### Questions

30