Introduction to Query Processing and Query Optimization

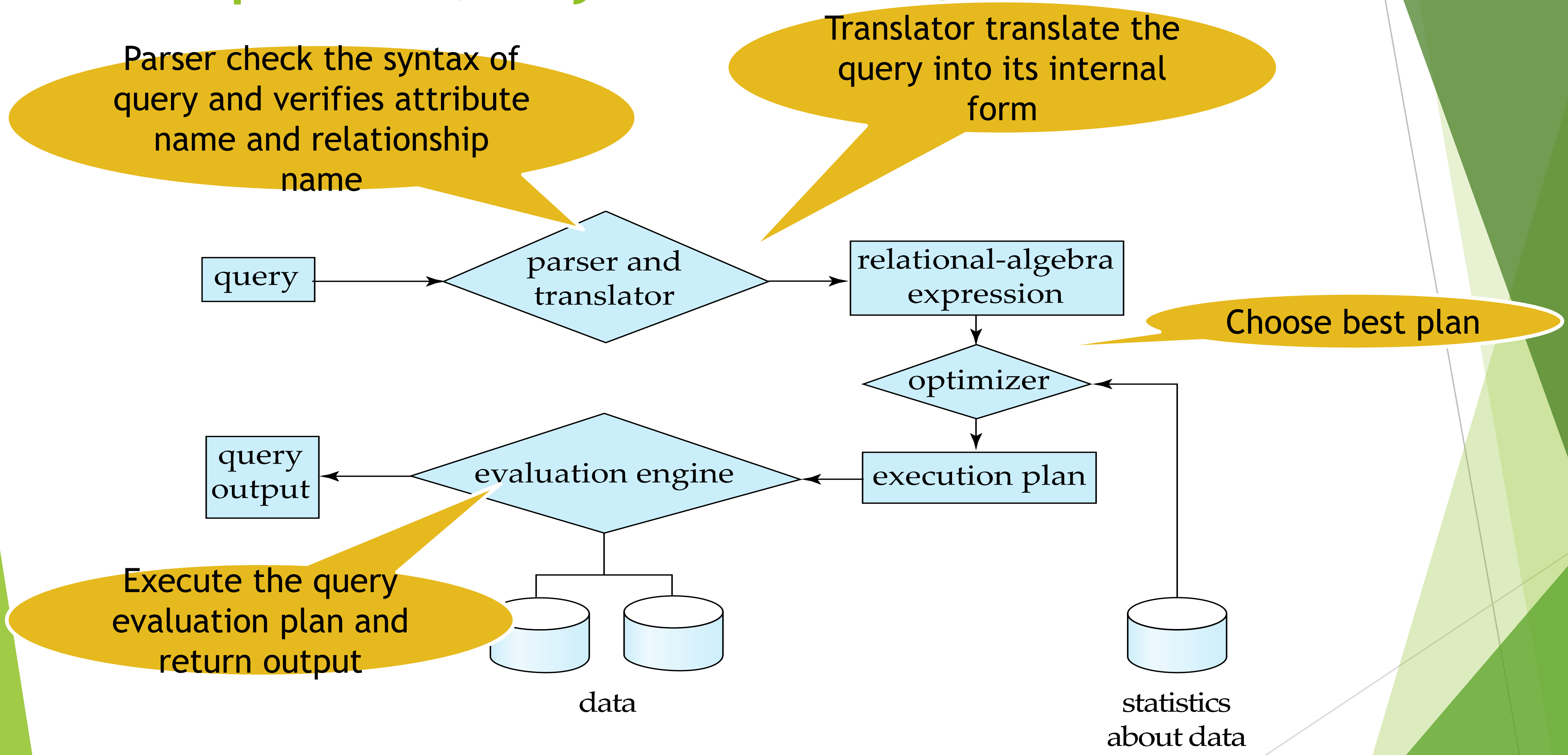
Outline

- Overview
- Measures of Query Cost
- Query Optimization

What is Query Processing?

- Query processing: Activities involved in extracting data from a database.
- Three basic steps:
  1. Parsing and Translation
  2. Optimization
  3. Evaluation

Steps in Query Processing



Measures of Query Cost

- Cost is generally measured as total elapsed time for answering query
- Many factors contribute to time cost
    1. Disk accesses (Time to process a data request and retrive data from the storage device)
    2. CPU (time to execute a query)
    3. Network communication cost
- Disk access is the predominant cost, and is also relatively easy to estimate.
- 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

- For simplicity we just use the number of block transfers from disk and the number of seeks as the cost measures

    tT – time to transfer one block

    tS – time for one seek

- Cost for b block transfers plus S seeks

    b * tT + S * tS

- 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

Select operation

- **Symbol**: σ
- **Notation**: σ condition (Relation)
- **Operation** : Select tuple from a relation that satisfy a given condition.

- Search algorithm

1. Linear search (A1)
2. Binary search (A2)

Linear search (A1)

- It Scan each file block and test all records to see whether they satisfy the selection condition.
  - Cost estimate = br block transfers
    - br denotes number of blocks containing records from relation r
  - If selection is on a key attribute (primary key), then system can stop on finding record
    - cost = (br /2) block transfers
  - Linear search can be applied regardless of
    - Selection condition or
    - Ordering of records in the file, or
    - Availability of indices
- This algorithm is slower than binary search algorithm.

Binary search (A2)

- Is used when selection is an **equality comparison** on the **primary key** and **relation is sorted on primary key attribute**.

- Cost of binary search = [log2(br)]

  br denotes number of blocks containing records from relation r

- If the selection is on **non primary attribute** then **multiple block may contains required records** , then the **cost of scanning** such block need to be **added** to the cost estimate.

- This algorithm is faster than linear search algorithm

Evaluation of expressions

- Method

1. Materialization
2. Pipelining

Materialization

- Materialized evaluation: evaluate one operation at a time, starting at the lowest-level. (from bottom and perform the inner most operations first)
- The intermediate results of each operation is materialized (store in temporary relation)and become input for subsequent(evaluate next-level operations).
- The cost of materialization is the sum of the individual operations plus the cost of writing the intermediate results to disk.
- The problem ;

1. Creates lots of temporary relation
2. Perform lots of I/O operation

Pipelining

- It evaluate several operations simultaneously, passing the results of one operation on to the next.
- To reduce number of intermediate temporary relations , we pass results of one operation to the next operation in the pipeline.
- Combining operations into a pipeline eliminates the cost of reading and writing temporary relations.
- Much cheaper than materialization: no need to store a temporary relation to disk.
- Pipelines can be executed in two ways:
- **Demand driven** –system makes request for tuples from the operation at the top of pipeline
- **Producer driven** – Operation do not wait for request to produce tuple but generate the tuples eagerly.

Query Optimization

Process of selecting the most efficient query evaluation plan

Query Optimization

Customer

Cid	Ano	C_name
101	A1	Ram
102	A2	Harsh
103	A3	Deepak
104	A4	Gopal

Account

Ano	Balance
A1	3000
A2	1000
A3	2000
A4	4000

$$\Pi_{Cust\_Name}(\underbrace{\sigma_{Balance<2500}(\text{account})}_{\text{2 records}} \bowtie \underbrace{\text{customer}}_{\text{4 records}})$$

Efficient Plan

$$\Pi_{Cust\_Name}(\sigma_{Balance<2500}(\underbrace{\text{Account}}_{\text{4 records}} \bowtie \underbrace{\text{Customer}}_{\text{4 records}}))$$

Transformation of relational Expression

Cascade of selection

Combined selection operation can be divided into sequence of individual selection.

Customer

Cid	Ano	Cust_name	Balance
C01	1	Raj	3000
C02	2	Meet	1000
C03	3	Harsh	2000
C04	4	Punit	4000

Output

Cid	Ano	Cust_name	Balance
C02	2	Meet	1000

$$\sigma_{Ano<3 \wedge Balance<2000} (Customer) = \sigma_{Ano<3} (\sigma_{Balance<2000} (Customer))$$

$$\sigma_{\theta 1 \wedge \theta 2} (E) = \sigma_{\theta 1}(\sigma_{\theta 2} (E))$$

Selection operation

Selection operation are commutative

Customer			
Cid	Ano	Cust_name	Balance
C01	1	Raj	3000
C02	2	Meet	1000
C03	3	Harsh	2000
C04	4	Punit	4000

Output			
Cid	Ano	Cust_name	Balance
C02	2	Meet	1000

$\sigma_{Ano<3}(\sigma_{Balance<2000}(Customer))$ = $\sigma_{Balance<2000}(\sigma_{Ano<3}(Customer))$

$$\sigma_{\theta 1}(\sigma_{\theta 2}(E) = \sigma_{\theta 2}(\sigma_{\theta 1}(E))$$

Project opeartion

- If more than one projection operation is used in expression then only the outer projection operation is required.

Customer

Cid	Ano	Cust_name	Balance
C01	1	Raj	3000
C02	2	Meet	1000
C03	3	Harsh	2000
C04	4	Punit	4000

Output

Cust_name
Raj
Meet
Harsh
Punit

$$\Pi_{Cust\_name}\left(\Pi_{Ano,\ Cust\_name}(Customer)\right) = \Pi_{Cust\_name}(Customer)$$

$$\Pi_{L1}(\Pi_{L2}(\ldots(\Pi_{Ln}(E))\ldots)) = \Pi_{L1}(E)$$

Join

- Natural join operations are associative

$$(E1 \bowtie E2) \bowtie E3 = E1 \bowtie (E2 \bowtie E3)$$

Union and Intersection

- Set operations union and intersection are commutative.
- Set operation union and intersection are associative.

Customer
Cust_name
Raj
Meet

Employee
Emp_name
Meet
Suresh

Student
Emp_name
Raj
Meet

Union

Output
Name
Raj
Meet
Suresh

Intersect

Output
Name
Meet

Example

Consider the Relational Algebra expression given below:
$(\mathtt{X} \bowtie_{\theta} \mathtt{Y}) \cap (\mathtt{X} \bowtie_{\theta} \mathtt{Z})$ where X, Y, and Z are relational algebra expressions.

Identify the correct equivalent Relational Algebra expression.

a) $\mathtt{X} \bowtie_{\theta} (\mathtt{Y} \bowtie_{\theta} \mathtt{Z})$

b) $(\mathtt{X} \bowtie_{\theta} \mathtt{Y}) - \mathtt{Z}$

c) $(\mathtt{X} \bowtie_{\theta} \mathtt{Y}) - (\mathtt{X} \bowtie_{\theta} \mathtt{Z})$

d) $\mathtt{X} \bowtie_{\theta} (\mathtt{Y} \cap \mathtt{Z})$

Join Operation

- Several different algorithms to implement joins
  - Nested-loop join
  - Block nested-loop join
  - Indexed nested-loop join
  - Merge-join
  - Hash-join
- Choice based on cost estimate

Nested-Loop Join

- To compute the theta join $r \bowtie_{\theta} s$

  for each tuple t_r **in** r **do begin**

  for each tuple t_s **in** s **do begin**

  test pair (t_r, t_s) to see if they satisfy the join condition θ

  if they do, add $t_r \bullet t_s$ to the result.

  end

  end
- r is called the **outer relation** and s the **inner relation** of the join.
- Requires no indices and can be used with any kind of join condition.
- Expensive since it examines every pair of tuples in the two relations.

Nested-Loop Join (Cont.)

- In the worst case, if there is enough memory only to hold one block of each relation, the estimated cost is
  $n_r * b_s + b_r$ block transfers, plus
  $n_r + b_r$ seeks
- N_{r-} is the number of records in relation r , br is the number of blocks in which they exist
- If the smaller relation fits entirely in memory, use that as the inner relation.
  - Reduces cost to $b_r + b_s$ block transfers and 2 seeks
- Examples use the following information
  - Number of records of *student*: 5,000 *takes*: 10,000
  - Number of blocks of *student*: 100 *takes*: 400
- Assuming worst case memory availability cost estimate is
  - with *student* as outer relation:
    - $5000 * 400 + 100 = 2{,}000{,}100$ block transfers,
    - $5000 + 100 = 5100$ seeks
  - with *takes* as the outer relation
    - $10000 * 100 + 400 = 1{,}000{,}400$ block transfers and 10,400 seeks
- If smaller relation (*student*) fits entirely in memory, the cost estimate will be 500 block transfers.

Recap

- Query processing
- Measures of Query Cost
- Evaluation of expressions
- Query representation