

# Query Optimization

## Chapter 15

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 1

---

---

---

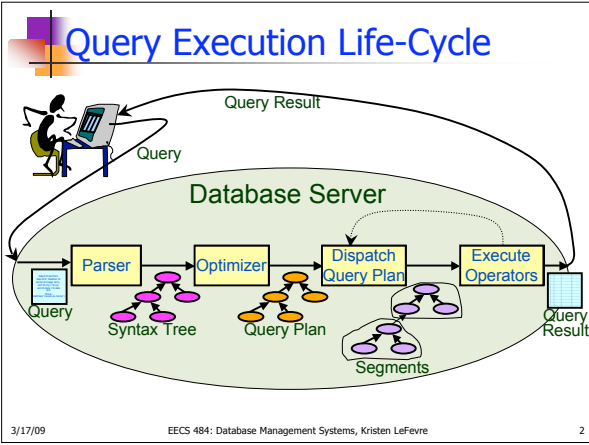
---

---

---

---

---




---

---

---

---

---

---

---

---

### Query Optimization

- Given a SQL query, how do we evaluate it efficiently?
- *Query Optimizer* – Important component of a DBMS
  - Convert SQL query *blocks* to extended relational algebra expressions
  - Enumerate alternative evaluation plans
  - Choose a plan based on estimated cost

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 3

---

---

---

---

---

---

---

---



## Query Evaluation Plan

- Annotated, extended RA tree
- Operator Interface: open(), getNext(), close()
- Intermediate Results (multiple ops):
  - **Pipelined:** Tuples resulting from one operator fed directly into the next
  - **Materialized:** Create a temporary table to store intermediate results

3/17/09

EECS 484: Database Management Systems, Kristen LeFevre

4

---

---

---

---

---

---

---

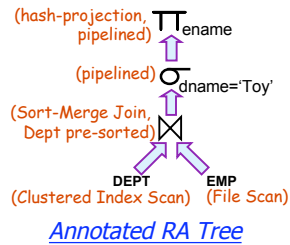
---



## Example

EMP (ssn, ename, addr, sal, did)  
 DEPT (did, dname, floor, mgr)

SELECT DISTINCT E.ename  
 FROM Emp E, Dept D  
 WHERE D.dname = 'Toy'  
 AND D.did = E.did



3/17/09

EECS 484: Database Management Systems, Kristen LeFevre

5

---

---

---

---

---

---

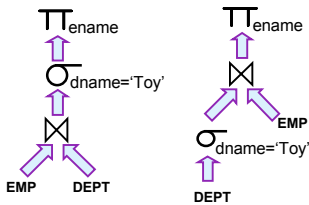
---

---



## Alternative Plans

- Can be many extended RA trees that produce the same result!



Also, different algorithms for each operator

3/17/09

EECS 484: Database Management Systems, Kristen LeFevre

6

---

---

---

---

---

---

---

---

## RA Equivalence

- $\sigma_{P_1}(\sigma_{P_2}(R)) \equiv \sigma_{P_2}(\sigma_{P_1}(R))$  ( $\sigma$  commutativity)
- $\sigma_{P_1 \wedge P_2 \dots \wedge P_n}(R) \equiv \sigma_{P_1}(\sigma_{P_2}(\dots \sigma_{P_n}(R)))$  (cascading  $\sigma$ )
- $\Pi_{a_1}(R) \equiv \Pi_{a_1}(\Pi_{a_2}(\dots \Pi_{a_k}(R)\dots))$ ,  $a_i \subseteq a_{i+1}$  (cascading  $\Pi$ )
- $R \bowtie S \equiv S \bowtie R$  (commutativity)
- $R \bowtie (S \bowtie T) \equiv (R \bowtie S) \bowtie T$  (associativity)
- $\sigma_P(R \bowtie S) \equiv (R \bowtie_P S)$  (if P is a join predicate)
- $\sigma_P(R \bowtie S) \equiv \sigma_{P_1}(\sigma_{P_2}(R) \bowtie_{P_4} \sigma_{P_3}(S))$   $P = P_1 \wedge P_2 \wedge P_3 \wedge P_4$
- $\Pi_{A_1, A_2, \dots, A_n}(\sigma_P(R)) \equiv \Pi_{A_1, A_2, \dots, A_n}(\sigma_P(\Pi_{A_1, \dots, A_n, B_1, \dots, B_M}(R)))$   
 $B_1 \dots B_M$  attributes in P

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 7

---

---

---

---

---

---

---

---

## Extended RA

HAVING<sub>MAX(SALARY) > z( ... )</sub>  
 GROUP BY<sub>D.did ( ... )</sub>

$\Pi_{did, \max(\text{salary})}$   
 ↑  
 HAVING count(\*) > 10  
 ↑  
 GROUP BY did  
 ↑  
 $\sigma_{\text{addr}='Ann Arbor'}$   
 ↑  
 EMP

Select E.did, Max (E.salary)  
 From Emp E  
 Where addr = 'Ann Arbor'  
 Group By E.did  
 Having count(\*) > 10

Simplification: Only optimize the s,  $\Pi$ , X  
 Project Group By/Having attributes  
 Choose from different aggregate algorithms

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 8

---

---

---

---

---

---

---

---

## Query Optimization – Main Issues

- Which (equivalent) plans do we consider?
- How do we estimate the cost of a plan?
- **Ideally:** Want to find best plan
- **Practically:** Avoid worst plans! Look at a subset of all plans
- Typical optimizations
  - Re-ordering joins
  - "Pushing" selections and projections

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 9

---

---

---

---

---

---

---

---

## System R Optimizer

- Most widely used currently; works well for < 10 joins
- **Cost estimation:** Approximate art at best
  - Catalog statistics
    - cost of operation
    - result size
  - Combination of CPU and I/O costs
- **Plan Space:**
  - Only *left-deep plans*
  - Avoid Cartesian products

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 10

---

---

---

---

---

---

---

---

## Cost Estimation

- Estimate *cost* of each operation in plan tree
  - Depends on input cardinalities
  - Algorithm cost (see previous lecture)
- Estimate *size of result*
  - Use information about the input relations
  - For selections and joins, assume independence of predicates
- We'll discuss the **System R** cost estimation approach
  - Very inexact, but works OK in practice
  - More sophisticated techniques known now

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 11

---

---

---

---

---

---

---

---

## Pricing Plans: Statistics

- Statistics stored in the catalogs
  - Relation
    - Cardinality (# rows)
    - Size in pages
  - Index
    - Cardinality (# distinct keys)
    - Size in pages
    - Height
    - Range
- Catalogs update periodically
  - Can be slightly inconsistent
- Commercial systems use histograms
  - More accurate estimates

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 12

---

---

---

---

---

---

---

---

## Size Estimation and Reduction Factors

```
SELECT attribute list
FROM relation list
WHERE term1 AND ... AND termk
```

Question: What is the cardinality of the result set?

- Max # tuples: product of input relation cardinalities
- Each term “filters” out some tuples: *Reduction factor*
- *Result cardinality* = Max # tuples \* product of all RF's.
- *Assumption: terms are independent!*
- Term *col=value* RF:  $1/NKeys(I)$ , given index I on col
- Term *col1=col2* RF:  $1/MAX(NKeys(I1), NKeys(I2))$
- Term *col>value* RF:  $(High(I)-value)/(High(I)-Low(I))$

3/17/09

EECS 484: Database Management Systems, Kristen LeFevre

13

---

---

---

---

---

---

---

---

## Plan Enumeration

- Two main cases:
  - Single-relation plans
  - Multiple-relation plans
- Single-relation plan (no joins). Access Plans:
  - file scan
  - index scan(s): Clustered, Unclustered
    - More than one index may “match” predicates
    - e.g. Clustered index I matching one or more selects:  
Cost:  $(NPages(I)+NPages(R)) * \text{product of RF's of matching selects.}$
  - Choose the one with the least estimated cost.
  - Merge/pipeline selection and projection (and aggregation)
    - RID intersection techniques
    - Index aggregate evaluation

3/17/09

EECS 484: Database Management Systems, Kristen LeFevre

14

---

---

---

---



---

---

---

---

## Example

EMP (ssn, ename, addr, sal, did)

```
SELECT E.ename
FROM Emp E
WHERE E.did=8
AND E.sal > 40K
```

- Index on did:
  - Tuples Retrieved:  $(1/10) * 10,000$
  - Clustered index:  $(1/10) * (100+1,000)$  pages
  - Unclustered index:  $(1/10) * (100+10,000)$  pages
- Index on sal:
  - Clustered index:  $(200-40)/(200-10) * (100+1,000)$  pages
  - Unclustered index: ...
- File scan: 1,000 pages

1,000 data pages, 10K tuples  
100 pages in B+-tree  
# depts: 10  
Salary Range: 10K – 200K

3/17/09

EECS 484: Database Management Systems, Kristen LeFevre

15

---

---

---

---

---

---

---

---

## Queries Over Multiple Relations

- System R: Only consider *left-deep* join trees
  - Used to restrict the search space
  - Left-deep plans can be *fully pipelined*.
    - Intermediate results not written to temporary files.
    - Not all left-deep trees are fully pipelined (e.g., SM join).

Linear Tree: at least 1 child in every join node is a base relation

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 16

---

---

---

---

---

---

---

---

## Enumeration of Left-Deep Plans

- Decide:
  - Join order
  - Join method for each join
- Enumerated using N passes (if N relations joined):
  - Pass 1: Find best 1-relation plan for each relation.
  - Pass 2: Find best way to join result of each 1-relation plan (as outer) to another relation. (*All 2-relation plans.*)
  - Pass N: Find best way to join result of a (N-1)-relation plan (as outer) to the N'th relation. (*All N-relation plans.*)
- For each subset of relations, retain only:
  - Cheapest plan overall, plus
  - Cheapest plan for each *interesting order* of the tuples.

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 17

---

---

---

---

---

---

---

---

## Example

$\Pi_{ename} \sigma_{dname = 'Toy'} (EMP \bowtie DEPT)$

EMP (ssn, ename, addr, sal, did) DEPT (did, dname, floor, mgr)

Pass 1: EMP: E1: S(EMP), E2: I(EMP.did)  
 Cost: 1000 1000+100 **KEEP E1 and E2!**

DEPT: D1: S(DEPT), D2: I.(DEPT.did), D3: I(DEPT.dname)  
 Cost: 50 50+5 3+5 **KEEP D2 and D3**

Pass 2: Consider EMP  $\bowtie$  DEPT and DEPT  $\bowtie$  EMP  
 EMP  $\bowtie$  DEPT, Alternatives:  
 1. E1  $\bowtie$  D2: Algorithms ...  
 2. E1  $\bowtie$  D3: Algorithms ...  
 3. E2  $\bowtie$  D2: Algorithms SM, NL, BNL, NL-IDX, Hash  
 4. E2  $\bowtie$  D3: Algorithms

Similarly consider DEPT  $\bowtie$  EMP  
 Pick cheapest 2-relation plan. Done (with join optimization)

**Next Consider GROUP BY (if present) ...**

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 18

---

---

---

---

---

---

---

---

## Enumeration of Plans (Contd.)

- ORDER BY, GROUP BY handled as a final step,
- Only "join" relations if there is a connecting join condition i.e., avoid Cartesian products if possible.
- This approach is still exponential in the # of tables.

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 19

---

---

---

---

---

---

---

---

## Summary

- Query optimization critical to the DBMS performance
  - Helps understand performance impact of database design
- Two parts to optimizing a query:
  - Enumerate alternative plans. (Typically only consider left-deep plans)
  - Estimate cost of each plan: size of result and cost of algorithm
    - Key issues: Statistics, indexes, operator implementations.
- Single-relation queries: Pick cheapest access plan + interesting order
- Multiple-relation queries:
  - All single-relation plans are first enumerated. Selections/projections considered as early as possible.
  - For each 1-relation plan, consider all ways of joining another relation (as inner)
  - Keep adding 1-relation plan until done
  - At each level, retain cheapest plan, and best plan for each interesting order

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 20

---

---

---

---

---

---

---

---

## Announcements

- Minirel Project 1, due Monday
- Quiz 2 (next Wednesday, last 30 minutes)
- Optional Exercises: 12.1 (all parts), 15.1, 15.5, 15.7, 15.9

3/17/09 EECS 484: Database Management Systems, Kristen LeFevre 21

---

---

---

---

---

---

---

---