Chapters 15 and 16: Query Processing

(Slides by Hector Garcia-Molina, http://www-db.stanford.edu/~hector/cs245/notes.htm)

Query Processing

Q → Query Plan

Focus: Relational System

Example

Select B,D
From R,S
Where R.A = "c"
AND S.E = 2
AND R.C=S.C

R:

A	B	C
a	1	10
b	1	20
c	2	10
d	2	35
e	3	45

S:

C	D	E
10	x	2
20	y	2
30	z	2
40	x	1
50	y	3

Answer

B	D
2	x

- How do we execute query?

One idea

- Do Cartesian product
- Select tuples
- Do projection

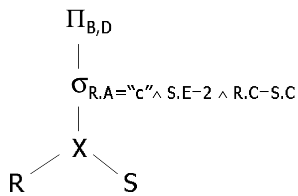
RXS

R.A	R.B	R.C	S.C	S.D	S.E
a	1	10	10	x	2
a	1	10	20	y	2
.					
.					
C	2	10	10	x	2
.					
.					

Bingo!
Got one...

Relational Algebra - can be used to describe plans...

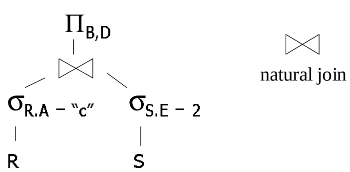
Ex: Plan I



OR: $\Pi_{B,D}\ [\ \sigma_{R.A=\text{"c"} \wedge S.E=2 \wedge R.C=S.C}\ (RXS)]$

Another idea:

Plan II



R

A	B	C
a	1	10
b	1	20
c	2	10
d	2	35
e	3	45

σ (R)

A	B	C
c	2	10

σ(S)

C	D	E
10	x	2
20	y	2
30	z	2

S

C	D	E
10	x	2
20	y	2
30	z	2
40	x	1
50	y	3

$\bowtie$

Plan III

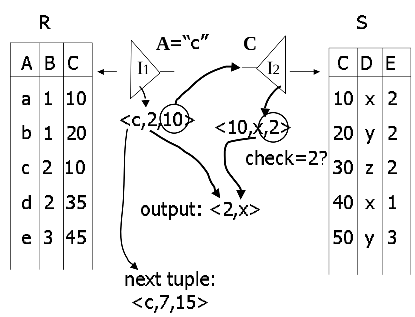
Use R.A and S.C Indexes

(1) Use R.A index to select R tuples with R.A = “c”

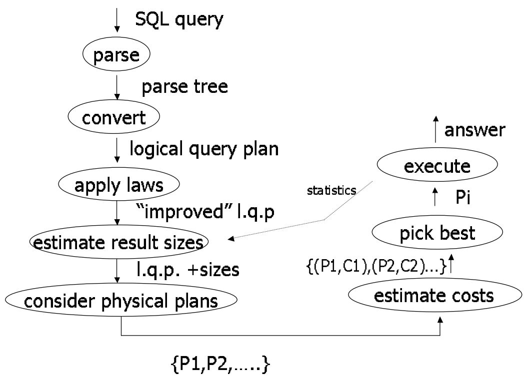
(2) For each R.C value found, use S.C index to find matching tuples

(3) Eliminate S tuples S.E $\neq$ 2

(4) Join matching R,S tuples, project B,D attributes and place in result



Overview of Query Optimization

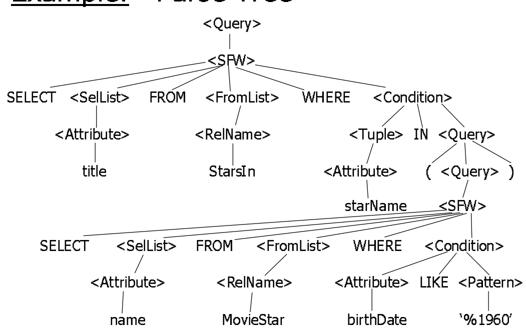


Example: SQL query

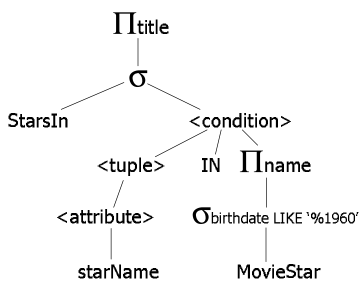
```
SELECT title
FROM StarsIn
WHERE starName IN (
        SELECT name
        FROM MovieStar
        WHERE birthdate LIKE '%1960'
);
```

(Find the movies with stars born in 1960)

Example: Parse Tree

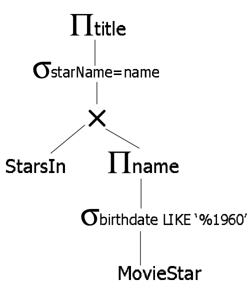


Example: Generating Relational Algebra



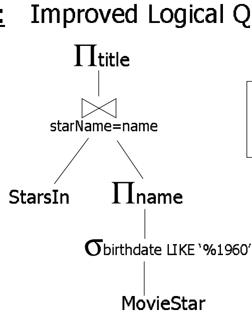
An expression using a two-argument σ, midway between a parse tree and relational algebra

Example: Logical Query Plan



Applying the rule for IN conditions

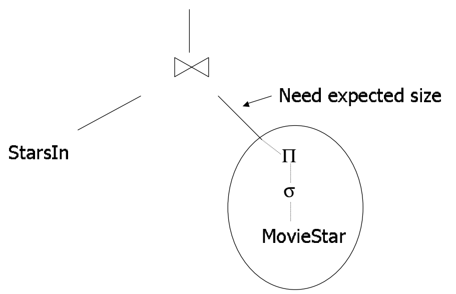
Example: Improved Logical Query Plan



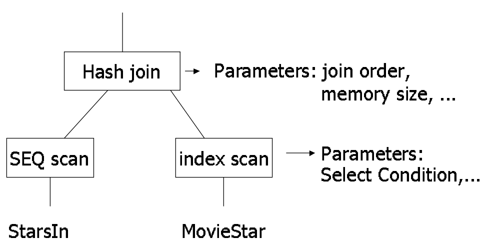
Question: Push project to StarsIn?

Fig. 7.20: An improvement on fig. 7.18.

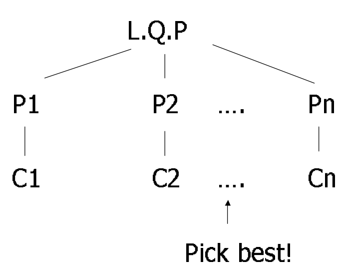
Example: Estimate Result Sizes



Example: One Physical Plan



Example: Estimate costs



Textbook outline

Chapter 5: Algebra for queries[bags vs sets]
- Select, project, join, ….[project list a,a+b->x,…]
- Duplicate elimination, grouping, sorting

Chapter 15:
- Physical operators
  - Scan,sort, …
- Implementing operators and estimating their cost

Chapter 16:

- Parsing
- Algebraic laws
- Parse tree -> logical query plan
- Estimating result sizes
- Cost based optimization

Query Optimization

- Relational algebra level
- Detailed query plan level
  - Estimate Costs
    - without indexes
    - with indexes
  - Generate and compare plans

Relational algebra optimization

- Transformation rules
  (preserve equivalence)
- What are good transformations?

Rules: Natural joins & cross products & union

$R \bowtie S \quad = \quad S \bowtie R$

$(R \bowtie S) \bowtie T \quad = R \bowtie (S \bowtie T)$

R x S = S x R

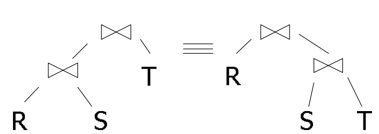
(R x S) x T = R x (S x T)

R U S = S U R

R U (S U T) = (R U S) U T

Note:

- Can also write as trees, e.g.:



Rules: Selects

$\sigma_{p1 \wedge p2}(R) = \sigma_{p1} [\sigma_{p2}(R)]$

$\sigma_{p1 \vee p2}(R) = [\sigma_{p1}(R)] \cup [\sigma_{p2}(R)]$

Rules: $\sigma + \bowtie$ combined

Let p = predicate with only R attribs
q = predicate with only S attribs

$\sigma_p (R \bowtie S) = [\sigma_p (R)] \bowtie S$

$\sigma_q (R \bowtie S) = R \bowtie [\sigma_q (S)]$

Which are "good" transformations?

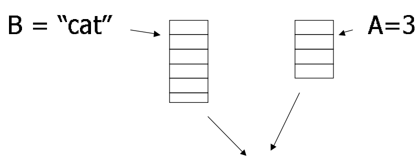
- □ $\sigma_{p1 \wedge p2} (R) \rightarrow \sigma_{p1} [\sigma_{p2} (R)]$
- □ $\sigma_p (R \bowtie S) \rightarrow [\sigma_p (R)] \bowtie S$
- □ $R \bowtie S \rightarrow S \bowtie R$

Conventional wisdom:
do projects early

Example: R(A,B,C,D,E)   x={E}
P: (A=3) $\wedge$ (B="cat")

$\pi_x \{\sigma_p\ (R)\}$ vs. $\pi_E \{\sigma_p \{\pi_{ABE}(R)\}\}$

$\boxed{\text{But}}$ What if we have A, B indexes?



Intersect pointers to get
pointers to matching tuples

Bottom line:

- No transformation is always good
- Usually good: early selections

In textbook: more transformations

- Eliminate common sub-expressions
- Other operations: duplicate elimination

Outline - Query Processing

- Relational algebra level
  - transformations
  - good transformations
- Detailed query plan level
  - estimate costs
  - generate and compare plans

- Estimating cost of query plan

(1) Estimating size of results
(2) Estimating # of IOs

Estimating result size

- Keep statistics for relation R
  - T(R) : # tuples in R
  - S(R) : # of bytes in each R tuple
  - B(R): # of blocks to hold all R tuples
  - V(R, A) : # distinct values in R for attribute A

Example

R

A	B	C	D
cat	1	10	a
cat	1	20	b
dog	1	30	a
dog	1	40	c
bat	1	50	d

A: 20 byte string
B: 4 byte integer
C: 8 byte date
D: 5 byte string

T(R) = 5 S(R) = 37
V(R,A) = 3 V(R,C) = 5
V(R,B) = 1 V(R,D) = 4

Size estimates for W = R1 x R2

T(W) = T(R1) $\times$ T(R2)

S(W) = S(R1) + S(R2)

Selection cardinality

SC(R,A) = average # records that satisfy equality condition on R.A

$$SC(R,A) = \begin{cases} \dfrac{T(R)}{V(R,A)} \\ \\ \dfrac{T(R)}{DOM(R,A)} \end{cases}$$

What about $W = \sigma_{z \geq val}(R)$?

$T(W) = ?$

- Solution # 1:
  $T(W) = T(R)/2$
- Solution # 2:
  $T(W) = T(R)/3$

Size estimate for $W = R1 \bowtie R2$

Let x = attributes of R1
y = attributes of R2

Case 1 $X \cap Y = \emptyset$

Same as R1 x R2

Case 2 $\quad W = R1 \bowtie R2 \qquad X \cap Y = A$

R1

A	B	C

R2

A	D

<u>Assumption:</u>

$V(R1,A) \leq V(R2,A) \Rightarrow$ Every A value in R1 is in R2
$V(R2,A) \leq V(R1,A) \Rightarrow$ Every A value in R2 is in R1

"containment of value sets"

- $V(R1,A) \leq V(R2,A) \quad T(W) = \dfrac{T(R2)\ T(R1)}{V(R2,A)}$
- $V(R2,A) \leq V(R1,A) \quad T(W) = \dfrac{T(R2)\ T(R1)}{V(R1,A)}$

[A is common attribute]

<u>In general</u> $\quad W = R1 \bowtie R2$

$$T(W) = \frac{T(R2)\ T(R1)}{\max\{ V(R1,A), V(R2,A) \}}$$

Using similar ideas,
we can estimate sizes of:

$\Pi_{AB}(R)$ …..

$\sigma_{A=a \wedge B=b}(R)$ ….

$R \bowtie S$ with common attribs. A,B,C

Union, intersection, diff, ….

Example:

$Z = R1(A,B) \bowtie R2(B,C) \bowtie R3(C,D)$

R1	T(R1) = 1000	V(R1,A)=50	V(R1,B)=100
R2	T(R2) = 2000	V(R2,B)=200	V(R2,C)=300
R3	T(R3) = 3000	V(R3,C)=90	V(R3,D)=500

Partial Result: $U = R1 \bowtie R2$

$T(U) = \frac{1000 \times 2000}{200}$

V(U,A) = 50
V(U,B) = 100
V(U,C) = 300

$Z = U \bowtie R3$

$T(Z) = \frac{1000 \times 2000 \times 3000}{200 \times 300}$

$V(Z,A) = 50$
$V(Z,B) = 100$
$V(Z,C) = 90$
$V(Z,D) = 500$

Summary

- Estimating size of results is an "art"
- Don't forget:
  Statistics must be kept up to date…
  (cost?)

Outline

- Estimating cost of query plan
  - Estimating size of results ← done!
  - Estimating # of IOs
- Generate and compare plans