Query Processing

Q → Query Plan

Focus: Relational System

• Others?

Example
Select B,D
From R,S
Where R.A = “c” ∧ S.E = 2 ∧ R.C=S.C

• How do we execute query?
- Do Cartesian product
- Select tuples
- Do projection

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

Ex: Plan I
\[ \Pi_{B,D} \sigma_{R.A = "c" \land S.E = 2 \land R.C = S.C} (R \times S) \]

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

Another idea:

Plan II
\[ \Pi_{B,D} \sigma_{R.A = "c" \land S.E = 2} (R \bowtie S) \]

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 ≠ 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
```
<Query>
<Attribute>
<title>
<RelName>
<Attribute>
<Tuple>
<Attribute>
<RelName>
WHERE
LIKE
'1960'
```

Example: Generating Relational Algebra
```
Π
σ
StarName = name
StarsIn
Π
name
σ
birthdate LIKE '%1960'
MovieStar

Fig. 16.14: An expression using a two-argument σ, midway between a parse tree and relational algebra.
```

Example: Improved Logical Query Plan
```
Π
σ
StarName = name
StarsIn
Π
name
σ
birthdate LIKE '%1960'
MovieStar

Fig. 16.21: The effect of query rewriting.

Question: Push project to StarsIn?
Example: Estimate Result Sizes

\[ \sigma_{J1}(\pi_{\text{StarsIn}}(\text{MovieStar})) \]

Need expected size

Example: One Physical Plan

Hash join
- Parameters: join order, memory size, project attributes, ...

SEQ scan
Index scan
- Parameters: Select Condition, ...

Example: Estimate costs

\[ \begin{align*}
\text{L} & : \text{L} \\
P1 & : P1 \\
P2 & : P2 \\
\vdots & : \vdots \\
Pn & : Pn \\
\text{C1} & : C1 \\
\text{C2} & : C2 \\
\vdots & : \vdots \\
\text{Cn} & : Cn \\
\end{align*} \]

Pick best!

Textbook outline

Chapter 15
15.1 Physical operators
- Scan, sort, ...
15.2-15.9 Implementing operators + estimating their cost

Chapter 16
16.1 Parsing
16.2 Algebraic laws
16.3 Parse tree -> logical query plan
16.4 Estimating result sizes
16.5-16.7 Cost based optimization

Reading textbook - Chapters 15,16
Optional: 15.7-15.9, 16.6-16.7
Optional: Duplicate elimination operator grouping, aggregation operators
Query Optimization - In class order

- 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

\[
R \bowtie S = S \bowtie R
\]
\[
(R \bowtie S) \bowtie T = R \bowtie (S \bowtie T)
\]

Note:

- Carry attribute names in results, so order is not important
- Can also write as trees, e.g.:

Rules: Natural joins & cross products & union

\[
R \bowtie S = S \bowtie R
\]
\[
(R \bowtie S) \bowtie T = R \bowtie (S \bowtie T)
\]

Rules: Selects

\[
\sigma_{p1 \land p2}(R) = \sigma_{p1} [ \sigma_{p2}(R) ]
\]
\[
\sigma_{p1 \lor p2}(R) = [ \sigma_{p1}(R) ] \cup [ \sigma_{p2}(R) ]
\]

Rules: Selects

\[
\sigma_{p1 \land p2}(R) = \sigma_{p1} [ \sigma_{p2}(R) ]
\]
\[
\sigma_{p1 \lor p2}(R) = [ \sigma_{p1}(R) ] \cup [ \sigma_{p2}(R) ]
\]
Bags vs. Sets

R = \{a,a,b,b,b,c\}
S = \{b,b,c,c,d\}
RUS = ?

- **Option 1** SUM
  - RUS = \{a,a,b,b,b,b,b,c,c,c,d\}
- **Option 2** MAX
  - RUS = \{a,a,b,b,b,c,c,d\}

```
Option 2 (MAX) makes this rule work:
σ_{p1 \land p2} (R) = σ_{p1} (R) \cup σ_{p2} (R)
```

Example:
R = \{a,a,b,b,b,c\}
P1 satisfied by a,b; P2 satisfied by b,c
σ_{p1 \land p2} (R) = \{a,a,b,b,b,c\}
σ_{p1} (R) = \{a,a,b,b,b\}
σ_{p2} (R) = \{b,b,b,c\}
σ_{p1} (R) \cup σ_{p2} (R) = \{a,a,b,b,b,c\}

```
“Sum” option makes more sense:

<table>
<thead>
<tr>
<th>Senators</th>
<th>Rep</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>T2</td>
</tr>
<tr>
<td>Yr</td>
<td>Yr</td>
</tr>
<tr>
<td>State</td>
<td>State</td>
</tr>
<tr>
<td>97</td>
<td>CA</td>
</tr>
<tr>
<td>99</td>
<td>CA</td>
</tr>
<tr>
<td>98</td>
<td>AZ</td>
</tr>
</tbody>
</table>
```

Union?

Executive Decision

-> Use “SUM” option for bag unions
-> Some rules cannot be used for bags

```
Rules: Project

Let: X = set of attributes
Y = set of attributes
XY = X \cup Y

π_{XY} (R) = \left( π_X (R) \right) \uplus π_Y (R)
```

```
Rules: σ_{p \land q} combined

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

σ_{p \land q} (R \uplus S) = [σ_{p} (R) \uplus \sigma_{q} (S)]
```
Rules: $\sigma + \pi$ combined (continued)

Some Rules can be Derived:

$\sigma_{paq} (R \bowtie q S) = \sigma$

$\sigma_{paqam} (R \bowtie q S) = \sigma$

$\sigma_{pvq} (R \bowtie q S) = \sigma$

Do one:

$\sigma_{paq} (R \bowtie q S) = [\sigma_p (R)] \bowtie q [\sigma_q (S)]$

$\sigma_{paqam} (R \bowtie q S) = \sigma_m [\sigma_p (R) \bowtie q (\sigma_q S)]$

$\sigma_{pvq} (R \bowtie q S) = [\sigma_p (R) \bowtie q S] \cup [R \bowtie q (\sigma_q S)]$

--> Derivation for first one:

$\sigma_{paq} (R \bowtie q S) = \sigma$

$\sigma_{paqam} (R \bowtie q S) = \sigma$

$\sigma_{pvq} (R \bowtie q S) = \sigma$

Rules: $\pi, \sigma$ combined

Let $x$ = subset of $R$ attributes

$z$ = attributes in predicate $P$ (subset of $R$ attributes)

$\pi_x \sigma_p (R) = \pi_x \{ \sigma_p [\pi_{xz'} (R)] \}$

Rules: $\pi, \pi$ combined

Let $x$ = subset of $R$ attributes

$y$ = subset of $S$ attributes

$z = \text{intersection of } R, S \text{ attributes}$

$\pi_{xy} (R \bowtie q S) = \pi_{xy} \{ \pi_{xz'} (R) \bowtie q \pi_{yz'} (S) \}$

$z' = z \cup \{ \text{attributes used in } P \}$

CS 277 - Spring 2002 Notes 6 37

CS 277 - Spring 2002 Notes 6 38

CS 277 - Spring 2002 Notes 6 39

CS 277 - Spring 2002 Notes 6 40

CS 277 - Spring 2002 Notes 6 41

CS 277 - Spring 2002 Notes 6 42
Rules for $\sigma$, $\pi$ combined with $X$

similar...
e.g., $\sigma_p (R \times S) = ?$

Rules $\sigma$, $U$ combined:

$\sigma_p (R \cup S) = \sigma_p (R) \cup \sigma_p (S)$
$\sigma_p (R - S) = \sigma_p (R) - S = \sigma_p (R) - \sigma_p (S)$

Which are “good” transformations?

- $\sigma_{p1 \land p2} (R) \rightarrow \sigma_{p1} [\sigma_{p2} (R)]$
- $\sigma_p (R \land S) \rightarrow [\sigma_p (R)] \land [\sigma_p (S)]$
- $R \land S \rightarrow S \land R$
- $\pi_x [\sigma_p (R)] \rightarrow \pi_x \{\sigma_p [\pi_x (R)]\}$

Conventional wisdom:
do projections early

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

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

But What if we have $A$, $B$ indexes?

B = "cat" $\rightarrow$

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

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat</td>
<td>1</td>
<td>10</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>cat</td>
<td>1</td>
<td>20</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>30</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>40</td>
<td>c</td>
<td></td>
</tr>
<tr>
<td>bat</td>
<td>1</td>
<td>50</td>
<td>d</td>
<td></td>
</tr>
</tbody>
</table>

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) x T(R2)
S(W) = S(R1) + S(R2)
Size estimate for $W = \sigma_{A=a}(R)$

$S(W) = S(R)$

$T(W) = ?$

Example

$$\begin{array}{cccc}
A & B & C & D \\
\text{cat} & 1 & 10 & a \\
\text{cat} & 1 & 20 & b \\
\text{dog} & 1 & 30 & a \\
\text{dog} & 1 & 40 & c \\
\text{bat} & 1 & 50 & d \\
\end{array}$$

$V(R,A) = 3$

$V(R,B) = 1$

$V(R,C) = 5$

$V(R,D) = 4$

$W = \sigma_{Z=\text{val}}(R)$

$T(W) = \frac{T(R)}{V(R,Z)}$

Assumption:

Values in select expression $Z = \text{val}$ are uniformly distributed over possible $V(R,Z)$ values.

Alternate Assumption:

Values in select expression $Z = \text{val}$ are uniformly distributed over domain with $\text{DOM}(R,Z)$ values.

Example

$$\begin{array}{cccc}
A & B & C & D \\
\text{cat} & 1 & 10 & a \\
\text{cat} & 1 & 20 & b \\
\text{dog} & 1 & 30 & a \\
\text{dog} & 1 & 40 & c \\
\text{bat} & 1 & 50 & d \\
\end{array}$$

$C=\text{val} \Rightarrow T(W) = (1/10)1 + (1/10)1 + ... = (5/10) = 0.5$

$B=\text{val} \Rightarrow T(W) = (1/10)5 + 0 + 0 = 0.5$

$A=\text{val} \Rightarrow T(W) = (1/10)2 + (1/10)2 + (1/10)1 = 0.5$
Example

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat</td>
<td>1</td>
<td>10</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>30</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>40</td>
<td>c</td>
<td></td>
</tr>
<tr>
<td>bat</td>
<td>1</td>
<td>50</td>
<td>d</td>
<td></td>
</tr>
</tbody>
</table>

Alternate assumption

\[ V(R,A) = 3 \quad \text{DOM}(R,A) = 10 \]
\[ V(R,B) = 1 \quad \text{DOM}(R,B) = 10 \]
\[ V(R,C) = 5 \quad \text{DOM}(R,C) = 10 \]
\[ V(R,D) = 4 \quad \text{DOM}(R,D) = 10 \]

W = \( \sigma_{z=\text{val}(R)} \)

T(W) = \( \frac{T(R)}{\text{DOM}(R,Z)} \)

Selection cardinality

\[ SC(R,A) = \text{average # records that satisfy equality condition on } R.A \]
\[ \frac{T(R)}{V(R,A)} \]

\[ SC(R,A) = \]
\[ \frac{T(R)}{\text{DOM}(R,A)} \]

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

T(W) = ?

- Solution # 1:
  T(W) = \( \frac{T(R)}{2} \)

- Solution # 2:
  T(W) = \( \frac{T(R)}{3} \)

Equivalently:

f \times V(R,Z) = \text{fraction of distinct values}

T(W) = \( \frac{[f \times V(Z,R)] \times T(R)}{V(Z,R)} \)

Solution # 3: Estimate values in range

Example

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Z</td>
<td>V(R,Z) = 10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W = ( \sigma_{z \geq 15} (R) )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\( f = \frac{20-15+1}{20-1+1} = \frac{6}{20} \) (fraction of range)

T(W) = \( f \times T(R) \)

Size estimate for W = R1 \times R2

Let x = attributes of R1

y = attributes of R2

Case 1

\( X \cap Y = \emptyset \)

Same as R1 x R2
Case 2  
\[ W = R_1 \bowtie R_2 \quad X \cap Y = A \]

\[
\begin{array}{c|c|c|c|c}
R_1 & A & B & C & R_2 & A & D \\
\end{array}
\]

Assumption:
\[ V(R_1,A) \leq V(R_2,A) \Rightarrow \text{Every A value in R}_1 \text{ is in R}_2 \]
\[ V(R_2,A) \leq V(R_1,A) \Rightarrow \text{Every A value in R}_2 \text{ is in R}_1 \]

"containment of value sets" Sec. 7.4.4

Computing \( T(W) \) when \( V(R_1,A) \leq V(R_2,A) \)

\[
\begin{array}{c|c|c|c|c|c}
R_1 & A & B & C & R_2 & A & D \\
\end{array}
\]

Take 1 tuple

Match

1 tuple matches with \( \frac{T(R_2)}{V(R_2,A)} \) tuples...

so \( T(W) = \frac{T(R_2) \times T(R_1)}{V(R_2,A)} \)

In general  
\[ W = R_1 \bowtie R_2 \]

\[
T(W) = \frac{T(R_2) \times T(R_1)}{\max\{V(R_1,A), V(R_2,A)\}}
\]

Case 2  with alternate assumption

Values uniformly distributed over domain

\[
\begin{array}{c|c|c|c|c}
R_1 & A & B & C & R_2 & A & D \\
\end{array}
\]

\[ \text{This tuple matches } T(R_2)/\text{DOM}(R_2,A) \text{ so} \]

\[
T(W) = \frac{T(R_2) \times T(R_1)}{\text{DOM}(R_2,A)} = \frac{T(R_2) \times T(R_1)}{\text{DOM}(R_1,A)}
\]

In all cases:

\[
S(W) = S(R_1) + S(R_2) - S(A). \quad \text{size of attribute A}
\]
Using similar ideas, we can estimate sizes of:

\[ \Pi_{A,B} (R) \] ... Sec. 16.4.2

\[ \sigma_{A=a, B=b} (R) \] ... Sec. 16.4.3

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

Sec. 16.4.5

Union, intersection, diff, ... Sec. 16.4.7

Note: for complex expressions, need intermediate T, S, V results.

E.g. \[ W = [\sigma_{A=a} (R1) ] \bowtie R2 \]

Treat as relation U

T(U) = T(R1)/V(R1,A) S(U) = S(R1)

Also need V(U, *) !

To estimate Vs

E.g., \[ U = \sigma_{A=a} (R1) \]

Say R1 has attribs A, B, C, D

V(U, A) = V(R1, A)
V(U, B) = V(R1, B)
V(U, C) = V(R1, C)
V(U, D) = V(R1, D)

Example

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat</td>
<td>1</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>cat</td>
<td>1</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>30</td>
<td>10</td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>40</td>
<td>30</td>
</tr>
<tr>
<td>bat</td>
<td>1</td>
<td>50</td>
<td>10</td>
</tr>
</tbody>
</table>

V(R1,A) = 3 V(R1,B) = 1 V(R1,C) = 5 V(R1,D) = 3

U = \[ \sigma_{A=a} (R1) \]

V(U,A) = 1 V(U,B) = 1 V(U,C) = \( \frac{T(R1)}{V(R1,A)} \)

V(D,U) ... somewhere in between

For Joins

\[ U = R1(A,B) \bowtie R2(A,C) \]

V(U,A) = min \{ V(R1, A), V(R2, A) \}
V(U,B) = V(R1, B)
V(U,C) = V(R2, C)

[called “preservation of value sets” in section 16.4.4]
Example:

\[ Z = R_1(A,B) \Join R_2(B,C) \Join R_3(C,D) \]

- \( R_1 \):
  - \( T(R_1) = 1000 \)
  - \( V(R_1,A) = 50 \)
  - \( V(R_1,B) = 100 \)

- \( R_2 \):
  - \( T(R_2) = 2000 \)
  - \( V(R_2,B) = 200 \)
  - \( V(R_2,C) = 300 \)

- \( R_3 \):
  - \( T(R_3) = 3000 \)
  - \( V(R_3,C) = 90 \)
  - \( V(R_3,D) = 500 \)

Partial Result: \( U = R \Join S \)

- \( T(U) = \frac{1000 \times 2000}{200} \)
- \( V(U,A) = 50 \)
- \( V(U,B) = 100 \)
- \( V(U,C) = 300 \)

Z = U \Join R_3

- \( T(Z) = \frac{1000 \times 2000 \times 3000}{200 \times 300} \)
- \( V(Z,A) = 50 \)
- \( V(Z,B) = 100 \)
- \( V(Z,C) = 90 \)
- \( V(Z,D) = 500 \)

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 I/Os next...
- Generate and compare plans