**Query Processing**

**Example**

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

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th></th>
<th>S</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td>10</td>
<td></td>
<td>10</td>
<td>x</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>20</td>
<td></td>
<td>20</td>
<td>y</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>2</td>
<td>10</td>
<td></td>
<td>30</td>
<td>z</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d</td>
<td>2</td>
<td>35</td>
<td></td>
<td>40</td>
<td>x</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>3</td>
<td>45</td>
<td></td>
<td>50</td>
<td>y</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Answer

<table>
<thead>
<tr>
<th>B</th>
<th>D</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Focus: Relational System

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

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

Another idea:

Plan II

\[
\Pi_{B,D} \\
\sigma_{R.A = "c"} \\
\sigma_{S.E = 2} \\
R \times S
\]

natural join

Bingo! Got one...

<table>
<thead>
<tr>
<th>R×S</th>
<th>R.A</th>
<th>R.B</th>
<th>R.C</th>
<th>S.C</th>
<th>S.D</th>
<th>S.E</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>a</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>x</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>a</td>
<td>1</td>
<td>10</td>
<td>20</td>
<td>y</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>2</td>
<td>10</td>
<td>10</td>
<td>x</td>
<td>2</td>
</tr>
</tbody>
</table>
### 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: Generating Relational Algebra

```
Πtitle
σ
StarsIn
<condition>
<tuple>
IN
Πname
<attribute>
σ
birthdate LIKE '%1960'
starName
MovieStar
```

An expression using a two-argument σ, midway between a parse tree and relational algebra.
Example: Logical Query Plan

\[
\Pi \text{title} \\
\sigma \text{starName=name} \\
\times \\
\Pi \text{name} \\
\sigma \text{birthdate LIKE '%1960'} \\
\text{MovieStar}
\]

Applying the rule for IN conditions

Example: Improved Logical Query Plan

\[
\Pi \text{title} \\
\text{starName=name} \\
\times \\
\Pi \text{name} \\
\sigma \text{birthdate LIKE '%1960'} \\
\text{MovieStar}
\]

Question: Push projection to StarsIn?

Example: Estimate Result Sizes

Example: One Physical Plan

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

Hash join

Parameters: Select Condition,...

SEQ scan

index scan

StarsIn

MovieStar
**Example: Estimate costs**

```
L Q P
/   |
P1   P2   ...   Pn
/     |
C1   C2   ...   Cn
```

Pick best!

**Outline**

**Algebra for queries**  
- Select, project, join, ...  
- Duplicate elimination, grouping, sorting

**Physical operators**  
- Scan, sort, ...

**Implementing operators**  
- Estimating their cost

**Query Optimization**

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

**Parsing**

Algebraic laws

Parse tree -> logical query plan

Estimating result sizes

Cost based optimization
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 (of attributes!) is not important
• Can also write as trees, e.g.:

\[
\begin{array}{c}
\bowtie \\
T
\end{array}
\quad \equiv 
\begin{array}{c}
\bowtie \\
S
\end{array} \bowtie \begin{array}{c}
\bowtie \\
T
\end{array}
\]

Rules: Natural joins & cross products & union

\[ R \bowtie S = S \bowtie R \]
\[ (R \bowtie S) \bowtie T = R \bowtie (S \bowtie T) \]
\[ R \times S = S \times R \]
\[ (R \times S) \times T = R \times (S \times T) \]
\[ R \cup S = S \cup R \]
\[ R \cup (S \cup T) = (R \cup S) \cup T \]
**Rules: Selects**

\[ \sigma_{p_1 \land p_2}(R) = \sigma_{p_1} [ \sigma_{p_2}(R)] \]

\[ \sigma_{p_1 \lor p_2}(R) = [ \sigma_{p_1}(R)] U [ \sigma_{p_2}(R)] \]

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

**Option 2 (MAX) makes this rule work:**

\[ \sigma_{p_1 \lor p_2}(R) = \sigma_{p_1}(R) U \sigma_{p_2}(R) \]

**Example:**  
R = \{a,a,b,b,b,c\}  
P1 satisfied by a,b;  P2 satisfied by b,c  
\[ \sigma_{p_1 \lor p_2}(R) = \{a,a,b,b,b,c\} \]

\[ \sigma_{p_1}(R) = \{a,a,b,b,b\} \]

\[ \sigma_{p_2}(R) = \{b,b,b,c\} \]

\[ \sigma_{p_1}(R) U \sigma_{p_2}(R) = \{a,a,b,b,b,c\} \]

**“Sum” option makes more sense:**

Senators (……)  Rep (……)  

T1 = \(\pi_{yr, state}\) Senators;  
T2 = \(\pi_{yr, state}\) Reps

<table>
<thead>
<tr>
<th>Yr</th>
<th>State</th>
<th>Yr</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>97</td>
<td>CA</td>
<td>99</td>
<td>CA</td>
</tr>
<tr>
<td>99</td>
<td>CA</td>
<td>98</td>
<td>CA</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 U Y

\[ \pi_{xy}(R) = \pi_x[\pi_y(R)] \]

Rules: \( \sigma + \bowtie \) combined

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

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

Rules: \( \sigma + \bowtie \) combined (continued)

Some Rules can be Derived:

\( \sigma_{p \land q}(R \bowtie S) = \)
\( \sigma_{p \land q \land m}(R \bowtie S) = \)
\( \sigma_{p \lor q}(R \bowtie S) = \)
Do one, others for homework:

\[ \sigma_{p \land q} (R \bowtie S) = [\sigma_p (R)] \bowtie [\sigma_q (S)] \]

\[ \sigma_{p \land q \land m} (R \bowtie S) = \]

\[ \sigma_m [ (\sigma_p R) \bowtie (\sigma_q S) ] \]

\[ \sigma_{p \lor q} (R \bowtie S) = \]

\[ [(\sigma_p R) \bowtie S] \cup [R \bowtie (\sigma_q S)] \]

→ Derivation for first one:

\[ \sigma_{p \land q} (R \bowtie S) = \]

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

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

\[ [\sigma_p (R)] \bowtie [\sigma_q (S)] \]

Rules: \( \pi, \sigma \) combined

Let \( x = \) subset of \( R \) attributes

\( z = \) attributes in predicate \( P \)

(\( \text{subset of R attributes} \))

\[ \pi_x [\sigma_p (R) ] = \pi_x [\sigma_p [ \pi_x (R) ] \]

Rules: \( \pi, \bowtie \) combined

Let \( x = \) subset of \( R \) attributes

\( y = \) subset of \( S \) attributes

\( z = \) intersection of \( R,S \) attributes

\[ \pi_{xy} (R \bowtie S) = \]

\[ \pi_{xy} [(\pi_{xz} (R) ) \bowtie (\pi_{yz} (S) )] \]
\[ \pi_{xy} \{ \sigma_p (R \bowtie S) \} = \]
\[ \pi_{xy} \{ \sigma_p [\pi_{xz}(R) \bowtie \pi_{yz}(S)] \} \]
\[ z' = z \cup \{ \text{attributes used in } P \} \]

**Rules** for \( \sigma, \pi \) combined with \( X \)

similar...

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

**Which are “good” transformations?**

\[ \emptyset \quad \sigma_{p1 \wedge p2} (R) \rightarrow \sigma_{p1} [\sigma_{p2} (R)] \]

\[ \emptyset \quad \sigma_p (R \bowtie S) \rightarrow [\sigma_p (R)] \bowtie S \]

\[ \emptyset \quad R \bowtie S \rightarrow S \bowtie R \]

\[ \emptyset \quad \pi_x [\sigma_p (R)] \rightarrow \pi_x \{ \sigma_p [\pi_{xz} (R)] \} \]
**Conventional wisdom:**

do projects early (always?)

*Example:* \( R(A,B,C,D,E) \quad x=\{E\} \)

\( P: (A=3) \land (B="\text{cat}") \)

\( \pi_x \{ \sigma_{p} (R) \} \quad \text{vs.} \quad \pi_E \{ \sigma_{p} \{ \pi_{ABE}(R) \} \} \)

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

\( B = "\text{cat}" \quad \triangleleft \quad A=3 \)

Intersect pointers to get pointers to matching tuples

**Bottom line:**

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

**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 I/Os

### 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

### Size estimates for \( W = R_1 \times R_2 \)

\[
T(W) = T(R_1) \times T(R_2) \\
S(W) = S(R_1) + S(R_2)
\]

---

**Example**

<table>
<thead>
<tr>
<th>R</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 estimate** for $W = \sigma_{A=a}(R)$

$S(W) = S(R)$

$T(W) = ?$

**Assumption:**

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

---

**Example**

<table>
<thead>
<tr>
<th>$R$</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>

$V(R,A)=3$

$V(R,B)=1$

$V(R,C)=5$

$V(R,D)=4$

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

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

**Alternate Assumption:**

Values in select expression $Z = \text{val}$ are uniformly distributed over domain with $\text{DOM}(R,Z)$ values.
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>a</td>
</tr>
<tr>
<td>cat</td>
<td>1</td>
<td>20</td>
<td>b</td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>30</td>
<td>a</td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>40</td>
<td>c</td>
</tr>
<tr>
<td>bat</td>
<td>1</td>
<td>50</td>
<td>d</td>
</tr>
</tbody>
</table>

Alternate assumption

\[
T(W) = \left( \frac{V(R,A)}{\text{DOM}(R,A)} \right) + \left( \frac{V(R,B)}{\text{DOM}(R,B)} \right) + \ldots
\]

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

T(W) = ?

\[
C=\text{val} \Rightarrow T(W) = \frac{1}{10} \cdot 1 + \frac{1}{10} \cdot 1 + \ldots
\]

= \( \frac{5}{10} = 0.5 \)

\[
B=\text{val} \Rightarrow T(W) = \frac{1}{10} \cdot 5 + 0 + 0 = 0.5
\]

\[
A=\text{val} \Rightarrow T(W) = \frac{1}{10} \cdot 2 + \frac{1}{10} \cdot 2 + \frac{1}{10} \cdot 1
\]

= 0.5

Selection cardinality

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

\[
SC(R,A) = \left\{ \frac{T(R)}{V(R,A)} \right\}
\]

\[
SC(R,A) = \frac{T(R)}{\text{DOM}(R,A)}
\]
What about $W = \sigma_{z \geq \text{val}(R)}$?

$T(W) = ?$

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

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

- Solution # 3: Estimate values in range

Example

<table>
<thead>
<tr>
<th>R</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min=1</td>
<td>V(R,Z)=10</td>
</tr>
<tr>
<td>Max=20</td>
<td>$W = \sigma_{z \geq 15}(R)$</td>
</tr>
</tbody>
</table>

$f = 20-15+1 = 6$ (fraction of range)

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

Equivalently:

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

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

Size estimate for $W = R_1 \times R_2$

Let $x = \text{attributes of } R_1$

$y = \text{attributes of } R_2$

Case 1

$X \cap Y = \emptyset$

Same as $R_1 \times R_2$
Case 2  \[ W = R_1 \Join R_2 \quad X \cap Y = A \]

\[ R_1 \quad A \quad B \quad C \quad R_2 \quad A \quad D \]

**Assumption:**

\[ V(R_1,A) \leq V(R_2,A) \implies \text{Every A value in } R_1 \text{ is in } R_2 \]
\[ V(R_2,A) \leq V(R_1,A) \implies \text{Every A value in } R_2 \text{ is in } R_1 \]

"containment of value sets"

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

Take 1 tuple

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**  \[ W = R_1 \Join R_2 \]

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

[A is common attribute]
**Case 2** with alternate assumption

Values uniformly distributed over domain

<table>
<thead>
<tr>
<th>R1</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R2</th>
<th>A</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This tuple matches \( T(R2)/\text{DOM}(R2, A) \) so

\[
T(W) = \frac{T(R2) T(R1)}{\text{DOM}(R2, A)} = \frac{T(R2) T(R1)}{\text{DOM}(R1, A)}
\]

In all cases:

\[
S(W) = S(R1) + S(R2) - S(A)_{\text{size of attribute A}}
\]

**Using similar ideas, we can estimate sizes of:**

\[
\Pi_{AB}(R) \quad \ldots
\]

\[
\sigma_{A=a \land B=b}(R) \quad \ldots
\]

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

Union, intersection, diff, ...

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

E.g. \( W = \left[ \sigma_{A=a}(R1) \right] \bowtie R2 \quad \text{Treat as relation U} \)

\[
T(U) = T(R1)/V(R1, A) \quad S(U) = S(R1)
\]

Also need \( V(U, *) \) !!
To estimate $V_S$

E.g., $U = \sigma_{A=a}(R_1)$

Say $R_1$ has attrs A,B,C,D

$V(U, A) = \quad V(U, B) = \quad V(U, C) = \quad V(U, D) =$ \[ \]

Example

<table>
<thead>
<tr>
<th>$R_1$</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>10</td>
<td></td>
</tr>
<tr>
<td>cat</td>
<td>1</td>
<td>20</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>30</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>dog</td>
<td>1</td>
<td>40</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>bat</td>
<td>1</td>
<td>50</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

$V(R_1, A) = 3 \quad V(R_1, B) = 1 \quad V(R_1, C) = 5 \quad V(R_1, D) = 3$

Possible Guess  $U = \sigma_{A=a}(R)$

$V(U, A) = 1 \quad V(U, B) = V(R, B)$

For Joins  $U = R_1(A,B) \bowtie R_2(A,C)$

$V(U, A) = \min \{ V(R_1, A), V(R_2, A) \}$

$V(U, B) = V(R_1, B)$

$V(U, C) = V(R_2, C)$

[“preservation of value sets”]
**Example:**

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

<table>
<thead>
<tr>
<th></th>
<th>( T(R_1) )</th>
<th>( V(R_1,A) )</th>
<th>( V(R_1,B) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>1000</td>
<td>50</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>( T(R_2) )</th>
<th>( V(R_2,B) )</th>
<th>( V(R_2,C) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>R2</td>
<td>2000</td>
<td>200</td>
<td>300</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>( T(R_3) )</th>
<th>( V(R_3,C) )</th>
<th>( V(R_3,D) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>R3</td>
<td>3000</td>
<td>90</td>
<td>500</td>
</tr>
</tbody>
</table>

**Partial Result:** \( U = R \bowtie S \)

\[
T(U) = \frac{1000 \times 2000 \times 3000}{200 \times 300} \\
V(U,A) = 50 \\
V(U,B) = 100 \\
V(U,C) = 300
\]

**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 ← occurs next...

• Generate and compare plans  ...Final step