CS 525: Advanced Database Organisation

09: Query Optimization - Logical

Boris Glavic

Slides: adapted from a course taught by Hector Garcia-Molina, Stanford InfoLab
SQL query → parse → parse tree → convert → logical query plan

apply laws → “improved” l.q.p

estimate result sizes → l.q.p. + sizes → consider physical plans

pick best → estimate costs → execute → answer

{(P1,C1),(P2,C2)...} → {(P1,P2,...)}
Query Optimization

- Relational algebra level
- Detailed query plan level
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?
  - Heuristic application of transformations
Query Equivalence

- Two queries $q$ and $q'$ are equivalent:
  - If for every database instance $I$
    - Contents of all the tables
    - Both queries have the same result

$$q \equiv q' \iff \forall I: q(I) = q'(I)$$
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.:

```
R   S   T  ≡  R   S   T
```

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

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

\[ \sigma_{p_1 \lor p_2}(R) = \]
Rules: Selects

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

\[ \sigma_{p_1 \lor p_2}(R) = [ \sigma_{p_1}(R)] \cup [ \sigma_{p_2}(R)] \]
Bags vs. Sets

\[ R = \{a, a, b, b, b, c\} \]
\[ S = \{b, b, c, c, d\} \]
\[ R \cup S = ? \]
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:

$$\sigma_{p_1 \lor p_2}(R) = \sigma_{p_1}(R) \cup \sigma_{p_2}(R)$$

Example: \(R=\{a,a,b,b,b,c\}\)

- \(P_1\) satisfied by \(a,b\)
- \(P_2\) satisfied by \(b,c\)
Option 2 (MAX) makes this rule work:

$$\sigma_{p_1 \lor p_2} (R) = \sigma_{p_1}(R) \cup \sigma_{p_2}(R)$$

Example: $R = \{a,a,b,b,b,c\}$

- $P_1$ satisfied by $a,b$; $P_2$ satisfied by $b,c$

  $$\sigma_{p_1 \lor p_2} (R) = \{a,a,b,b,b,c\}$$

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

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

  $$\sigma_{p_1}(R) \cup \sigma_{p_2}(R) = \{a,a,b,b,b,c\}$$
“Sum” option makes more sense:

<table>
<thead>
<tr>
<th>Senators (......)</th>
<th>Rep (......)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1 = \pi_{yr,\text{state}} \text{Senators}$;</td>
<td>$T_2 = \pi_{yr,\text{state}} \text{Reps}$</td>
</tr>
<tr>
<td>$T_1$</td>
<td>$T_2$</td>
</tr>
<tr>
<td>Yr</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  
$XY = X \cup Y$

$$\pi_{xy} (R) =$$
Rules: Project

Let: X = set of attributes
    Y = set of attributes
    XY = X U Y

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

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

\[
\pi_{xy}(R) = \pi_x[\pi_y(R)]
\]
Rules: $\sigma + \bowtie$ combined

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

$\sigma_p (R \bowtie S) =$

$\sigma_q (R \bowtie S) =$
Rules: $\sigma + \bowtie$ combined

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

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

$\sigma_q (R \bowtie S) = R \bowtie [\sigma_q (S)]$
Some Rules can be Derived:

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

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

\[ \sigma_{p \vee q} (R \bowtie S) = \]
Do one:

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

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

\[ \sigma_{p \lor q} (R \bowtie S) = \left[ (\sigma_p R) \bowtie S \right] \cup \left[ R \bowtie (\sigma_q S) \right] \]
--> 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 \( P \)

(subset of \( R \) attributes)

\[ \pi_x[\sigma_p(R)] = \]
Rules: \( \pi, \sigma \) combined

Let \( x = \) subset of \( R \) attributes
\[ z = \) attributes in predicate \( P \)
(subset of \( R \) attributes)

\[ \pi_x[\sigma_p (R)] = \{ \sigma_p [\pi_x (R)] \} \]
**Rules:** $\pi, \sigma$ combined

Let $x = \text{subset of } R \text{ attributes}$
$z = \text{attributes in predicate } P$
(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

\[ \pi_{xy} (R \bowtie S) = \]
Rules: $\pi$, $\bowtie$ combined

Let $x =$ subset of $R$ attributes
$y =$ subset of $S$ attributes
$z =$ intersection of $R,S$ attributes

$$\pi_{xy} (R \bowtie S) =$$

$$\pi_{xy}\{[\pi_{xz} (R)] \bowtie [\pi_{yz} (S)]\}$$
\[ \pi_{xy} \{ \sigma_p (R \bowtie S) \} = \]
\[ \pi_{xy} \left\{ \sigma_p (R \bowtie S) \right\} = \]

\[ \pi_{xy} \left\{ \sigma_p \left[ \pi_{xz'} (R) \bowtie \pi_{yz'} (S) \right] \right\} \]

\[ z' = z \cup \{ \text{attributes used in } P \} \]
Rules for $\sigma, \pi$ combined with $X$

similar...

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

- $\sigma_{p_1 \land p_2} (R) \rightarrow \sigma_{p_1} [\sigma_{p_2} (R)]$
- $\sigma_{p} (R \bowtie S) \rightarrow [\sigma_{p} (R)] \bowtie S$
- $R \bowtie S \rightarrow S \bowtie R$
- $\pi_x [\sigma_{p} (R)] \rightarrow \pi_x \{\sigma_{p} [\pi_{xz} (R)]\}$
Conventional wisdom:  
do projects early

Example: \( R(\mathbf{A,B,C,D,E}) \) \( x=\{E\} \)
\[
P: (A=3) \land (B=\text{"cat"})
\]

\[
\pi_x \{ \sigma_p (R) \} \quad \text{vs.} \quad \pi_E \{ \sigma_p \{ \pi_{ABE}(R) \} \}
\]
But What if we have A, B indexes?

B = “cat” → Intersect pointers to get pointers to matching tuples e.g., using bitmaps

A=3

Intersect pointers to get pointers to matching tuples e.g., using bitmaps
Bottom line:

- No transformation is always good
- Usually good: early selections
  - Exception: expensive selection conditions
  - E.g., UDFs
More transformations

- Eliminate common sub-expressions
- Detect constant expressions
- Other operations: duplicate elimination
Pushing Selections

• Idea:
  – Join conditions equate attributes
  – For parts of algebra tree (scope) store which attributes have to be the same
    • Called Equivalence classes

• Example: \( R(a,b), S(c,d) \)

\[
\sigma_{b=3} (R \bowtie_{b=c} S) = \sigma_{b=3} (R) \bowtie_{b=c} \sigma_{c=3} (S)
\]
Outer-Joins

- Not commutative
  - $R \bowtie S \neq S \bowtie R$

- $p$ – condition over attributes in $A$

- A list of attributes from $R$

\[ \sigma_p (R \bowtie_{A=B} S) \equiv \sigma_p (R) \bowtie_{A=B} S \]

Not \[ \sigma_p (R \bowtie_{A=B} S) \equiv R \bowtie_{A=B} \sigma_p (S) \]
Summary Equivalences

- **Associativity**: \((R \odot S) \odot T \equiv R \odot (S \odot T)\)
- **Commutativity**: \(R \odot S \equiv S \odot R\)
- **Distributivity**: \((R \odot S) \otimes T \equiv (R \otimes T) \odot (S \otimes T)\)
- **Difference between Set and Bag Equivalences**
- **Only some equivalence are useful**
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
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>

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,B) = 1
V(R,C) = 5
V(R,D) = 4
Size estimates for $W = R1 \times R2$

$T(W) =$

$S(W) =$
Size estimates for $W = R_1 \times R_2$

$T(W) = T(R_1) \times T(R_2)$

$S(W) = S(R_1) + S(R_2)$
Size estimate for $W = \sigma_{A=a} (R)$

$S(W) = S(R)$

$T(W) = ?$
**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>

\[ W = \sigma_{z = \text{val}}(R) \quad T(W) = \]

\[ V(R,A) = 3 \]
\[ V(R,B) = 1 \]
\[ V(R,C) = 5 \]
\[ V(R,D) = 4 \]
**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>

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

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

\[
\begin{align*}
V(R,A) &= 3 \\
V(R,B) &= 1 \\
V(R,C) &= 5 \\
V(R,D) &= 4
\end{align*}
\]
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

<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>

Alternate assumption

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

\[ W = \sigma_{z=val}(R) \quad T(W) = ? \]
C=val ⇒ T(W) = (1/10)1 + (1/10)1 + ... 
  = (5/10) = 0.5

B=val ⇒ T(W) = (1/10)5 + 0 + 0 = 0.5

A=val ⇒ T(W) = (1/10)2 + (1/10)2 + (1/10)1 
  = 0.5
Example

\[ W = \sigma_{z=\text{val}(R)}(R) \quad T(W) = \frac{T(R)}{\text{DOM}(R,Z)} \]
Selection cardinality

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

\[ SC(R,A) = \begin{cases} \frac{T(R)}{V(R,A)} & \text{if } V(R,A) \neq 0 \\ \frac{T(R)}{DOM(R,A)} & \text{if } DOM(R,A) \neq 0 \end{cases} \]
What about \( W = \sigma_{z \geq \text{val} (R)} \)?

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

\[ T(W) = ? \]

- Solution #1:
  \[ T(W) = \frac{T(R)}{2} \]
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></td>
<td>Min=1 $V(R,Z)=10$</td>
</tr>
<tr>
<td></td>
<td>Max=20 $W=\sigma_{z \geq 15}(R)$</td>
</tr>
</tbody>
</table>
• Solution # 3: Estimate values in range

Example | R | Z
--- | --- | ---
Min=1 | V(R,Z)=10
Max=20

\[ W = \sigma_{Z \geq 15} (R) \]

\[ f = \frac{20-15+1}{20-1+1} = \frac{6}{20} \]

\[ (\text{fraction of range}) \]

\[ T(W) = f \times T(R) \]
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 $W = R_1 \Join R_2$

Let $x =$ attributes of $R_1$
$y =$ attributes of $R_2$
Size estimate for $W = R1 \Join R2$

Let $x =$ attributes of $R1$

$y =$ attributes of $R2$

Case 1

$$X \cap Y = \emptyset$$

Same as $R1 \times R2$
Case 2 \[ W = R_1 \Join R_2 \quad X \cap Y = A \]
Case 2

\[ W = \text{R1} \bowtie \text{R2} \quad X \cap Y = A \]

\begin{array}{|c|c|c|c|}
\hline
\text{R1} & \text{A} & \text{B} & \text{C} \\
\hline
\text{R2} & \text{A} & \text{D} & \text{A} \\
\hline
\end{array}

**Assumption:**

\[ V(\text{R1},A) \leq V(\text{R2},A) \implies \text{Every A value in R1 is in R2} \]

\[ V(\text{R2},A) \leq V(\text{R1},A) \implies \text{Every A value in R2 is in R1} \]
Computing $T(W)$ when $V(R1,A) \leq V(R2,A)$

<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>

Take 1 tuple

Match
Computing $T(W)$ when $V(R1,A) \leq V(R2,A)$

Take 1 tuple

1 tuple matches with $\frac{T(R2)}{V(R2,A)}$ tuples...

So $T(W) = \frac{T(R2)}{V(R2,A)} \times T(R1)$
• \( V(R_1, A) \leq V(R_2, A) \)

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

• \( V(R_2, A) \leq V(R_1, A) \)

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

[A is common attribute]
In general $W = R_1 \boxtimes R_2$

$T(W) = \frac{T(R_2) \cdot T(R_1)}{\max\{V(R_1,A), V(R_2,A)\}}$
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>
</table>

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

$$T(W) = \frac{T(R2) \cdot T(R1)}{\text{DOM}(R2, A)} = \frac{T(R2) \cdot T(R1)}{\text{DOM}(R1, A)}$$

Assume the same
In all cases:

\[ S(W) = S(R1) + S(R2) - S(A) \]

size of attribute A
Using similar ideas, we can estimate sizes of:

\[ \Pi_{AB} \left( R \right) \]

\[ \sigma_{A=a \land B=b} \left( R \right) \]

\[ R \bowtie S \] with common attributes \( A, B, C \)

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

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

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 attributes \( A, B, C, D \)

\[
\begin{align*}
V(U, A) &= \\
V(U, B) &= \\
V(U, C) &= \\
V(U, D) &=
\end{align*}
\]
Example

R1

<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>

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

$U = \sigma_{A=a}(R1)$
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
Possible Guess \( U = \sigma_{A=a}(R) \)

\[
V(U,A) = 1 \\
V(U,B) = V(R,B)
\]
For Joins \[ U = R_1(A, B) \bowtie R_2(A, C) \]

\[
\begin{align*}
V(U, A) &= \min \{ V(R_1, A), V(R_2, A) \} \\
V(U, B) &= V(R_1, B) \\
V(U, C) &= V(R_2, C)
\end{align*}
\]
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>
<th>( T(R_2) )</th>
<th>( V(R_2,B) )</th>
<th>( V(R_2,C) )</th>
<th>( T(R_3) )</th>
<th>( V(R_3,C) )</th>
<th>( V(R_3,D) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>1000</td>
<td>50</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R2</td>
<td>2000</td>
<td>200</td>
<td>300</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R3</td>
<td>3000</td>
<td>90</td>
<td>500</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Partial Result: \[ U = R_1 \otimes R_2 \]

\[
\begin{align*}
T(U) &= \frac{1000 \times 2000}{200} \\
V(U,A) &= 50 \\
V(U,B) &= 100 \\
V(U,C) &= 300
\end{align*}
\]
\[ Z = U \otimes R3 \]

\[ T(Z) = \frac{1000 \times 2000 \times 3000}{200 \times 300} \quad V(Z,A) = 50 \]

\[ V(Z,B) = 100 \]

\[ V(Z,C) = 90 \]

\[ V(Z,D) = 500 \]
Approximating Distributions

- Summarize the distribution
  - Used to better estimate result sizes
  - Without the need to look at all the data

- Concerns
  - Error metric: How to measure preciseness
  - Memory consumption
  - Computational Complexity
Approximating Distributions

• Parameterized distribution
  – E.g., gauss distribution
  – Adapt parameters to fit data

• Histograms
  – Divide domain into ranges (buckets)
  – Store the number of tuples per bucket

• Both need to be maintained
Maintaining Statistics

• Use separate command that triggers statistics collection
  – Postgres: ANALYZE

• During query processing
  – Overhead for queries

• Use Sampling?
Estimating Result Size using Histograms

\[ \sigma_{A=\text{val}(R)} = ? \]

number of tuples in \( R \) with \( A \) value in given range
Estimating Result Size using Histograms

- $\sigma_{A=\text{val}}(R) = ?$
- $|B|$ - number of values per bucket
- $\#B$ – number of records in bucket

$$\frac{\#B}{|B|}$$
Join Size using Histograms

- \( R \Join S \)
- Use

\[
T(W) = \frac{T(R2) \cdot T(R1)}{\max\{V(R1,A), V(R2,A)\}}
\]

- Apply for each bucket
Join Size using Histograms

- \( V(R1,A) = V(R2,A) = \text{bucket size } |B| \)

\[
T(W) = \sum_{\text{buckets}} \frac{|B(R2)| \cdot |B(R1)|}{|B|}
\]
Equi-width vs. Equi-depth

- **Equi-width**
  - All buckets contain the same number of values
  - Easy, but inaccurate

- **Equi-depth (used by most DBMS)**
  - All buckets contain the same number of tuples
  - Better accuracy, need to sort data to compute
Equi-width vs. Equi-depth
Construct Equi-depth Histograms

- Sort input
- Determine size of buckets
  - \#bucket / \#tuples
- Example 3 buckets
  1, 5, 44, 6, 10, 12, 3, 6, 7
  1, 3, 5, 6, 6, 7, 10, 12, 44
  [1–5] [6–8] [9–44]
Advanced Techniques

• Wavelets
• Approximate Histograms
• Sampling Techniques
• Compressed Histograms
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 next...
  – Operator Implementations

• Generate and compare plans