CS522 Advanced Database Systems

Query Optimization

Chengyu Sun
California State University, Los Angeles

SQL Query Example

R

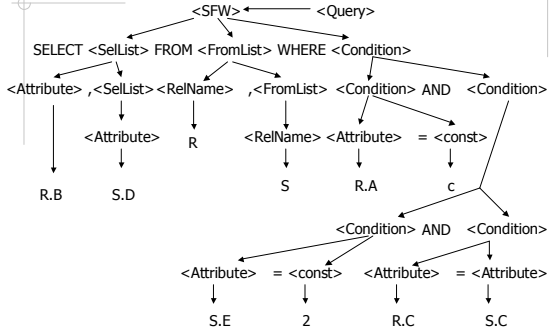
A	B	C
a	1	10
b	1	20
c	2	10
d	2	25
e	3	45

S

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

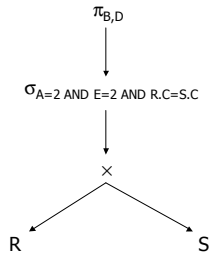
```
select B, D
    from R, S
    where R.A = c and S.E = 2 and R.C = S.C;
```

Parse Tree



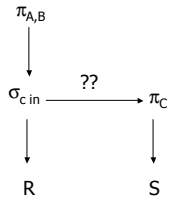
Logical Query Plan

- Operators
  - π, σ, ×
- Arguments (Operands)
  - relations
    - R and S
- Parameters
  - non-relations
    - B,D
    - A=2 AND E=2 AND R.C=S.C
- *Parameters can be "applied" to each tuple in the relation(s)*



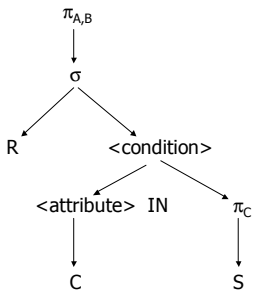
Subqueries in Conditions

```
select A,B from R where C in (select C from S);
```



- notational complications
- expensive to evaluate

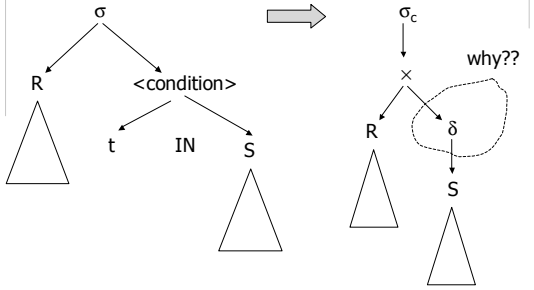
Two-argument Selection



Convert to Relational Algebra Selection

- Have to come up rules on a case-by-case basis
  - IN, EXISTS, ANY, ALL ...
  - correlated or uncorrelated
- In some rare cases we can leave the two-argument selection as part of the logical query plan

Example: Uncorrelated IN



Improve Logical Query Plans

- Transform a logical query plan to an equivalent one that is likely to lead to a more efficient physical plan

Simple Algebraic Laws for Transformation

- Commutative law
  - $R \cup S = S \cup R$
  - $R \cap S = S \cap R$
  - $R \times S = S \times R$
  - $R \triangleright\triangleleft S = S \triangleright\triangleleft R$
- Associative law
  - $(R \cup S) \cup T = R \cup (S \cup T)$
  - $(R \cap S) \cap T = R \cap (S \cap T)$
  - $(R \times S) \times T = R \times (S \times T)$
  - $(R \triangleright\triangleleft S) \triangleright\triangleleft T = R \triangleright\triangleleft (S \triangleright\triangleleft T)$

Bags vs. Sets

- Consider the distributive laws
  - $R \cap (S \cup T) = (R \cap S) \cup (R \cap T)$
  - $R \cup (S \cap T) = (R \cup S) \cap (R \cup T)$

Proofs HOWTO

- Prove set equivalence
  - $t \in A \rightarrow t \in B$
  - $t \in B \rightarrow t \in A$
- Prove bag equivalence
- Disprove

Selection Laws ...

- *Splitting*
  - $\sigma_{c1 \text{ AND } c2}(R) = \sigma_{c1}(\sigma_{c2}(R))$
  - $\sigma_{c1 \text{ OR } c2}(R) = \sigma_{c1}(R) \cup \sigma_{c2}(R)$
    - $\cup_S$ or $\cup_B$??

... Selection Laws

- *Pushing*
  - *op*: $\cup$, $\cap$, $-$, $\times$, $\triangleright\triangleleft$
  - push to both arguments
    - $\sigma_c (R \; op \; S) = \sigma_c(R) \; op \; \sigma_c(S)$
    - when??
  - push to one of the arguments
    - $\sigma_c (R \; op \; S) = \sigma_c(R) \; op \; S$
    - $\sigma_c (R \; op \; S) = R \; op \; \sigma_c(S)$
    - when??

Examples: Pushing Selections

- R(a,b) and S(b,c)
  - $\sigma_{(a=1 \text{ OR } a=3) \text{ AND } b<c} (R \triangleright\triangleleft S)$
  - $\sigma_{b=1}(R) \triangleright\triangleleft S$

Projection Laws

- *Adding* projections
- In general, we can project out attributes that are not *used* later on
- Examples:
  - R(a,b,c) and S(c,d,e)
    - $\pi_{a+e->x,b->y}(R \triangleright\triangleleft S)$
    - $\pi_{a+b->x,d+e->y}(R \triangleright\triangleleft S)$
  - Union, intersection, difference??

Project Law Examples

- Prove
  - $\pi_L(R \cup_B S) = \pi_L(R) \cup_B \pi_L(S)$
- Disprove
  - $\pi_L(R -_B S) = \pi_L(R) -_B \pi_L(S)$
  - $\pi_L(R -_s S) = \pi_L(R) -_s \pi_L(S)$

Some Other Laws

- Duplicate elimination
  - $\delta(R) = R$ if ...??
  - $\delta(R \times S) = \delta(R) \times \delta(S)$
  - $\delta(R \triangleright\triangleleft S) = \delta(R) \triangleright\triangleleft \delta(S)$
  - $\delta(\sigma_c(R)) = \sigma_c(\delta(R))$
- Group by
  - *Duplicate-impervious* aggregations

About Algebraic Laws

- There are too many to remember
- You can come up with more (as long as you can prove)
- Beware of the different semantics of sets and bags

Cost-based Query Optimization

- Choose the best logical or physical query plan
- What influence the "cost" of a query?
  - Choice of operators
  - Order of operators
  - Interaction between operators

Selectivity Estimation

- Selectivity = |ResultSet| / |DataSet|
- Cost estimation for logical query plans
  - All equivalent plans produce the same *final* result set
  - The plan which produces the smallest *intermediate* result set wins
- Provide information for choosing physical query plans

Selectivity Estimation with Simple Statistics

- T(R) – number of tuples in R
- V(R,a) – number of distinct values of attribute a
- V(R, $[a_1,a_2,...,a_n]$)

Estimating Selection Selectivity

- a=x: 1/V(R,a)
- a>x: ½ or 1/3
- a≠x: ??
- c_1 AND c_2: ??
- c_1 OR c_2: ??
- Example: R(a,b)
  - T(R) = 10000, V(R,a) = 50
  - Estimate | $\sigma_{a=10 \text{ OR } b<20}(R)$ |

Estimating Join Size ...

- Very hard problem even with more sophisticated methods
- Consider natural join of R(X,Y) and S(Y,Z)
  - 0
  - |R| or |S|
  - |R|*|S|

... Estimating Join Size ...

- Simplifying assumptions
  - Containment of value sets
    - if V(R,Y) <= V(R,Y), then every Y-value of R is a Y-value of S
  - Preservation of value sets
    - if A is an attribute of R but not a join attribute, then V(S ⋈ R, A) = V(R,A)
- When do these assumptions hold?

... Estimating Join Size

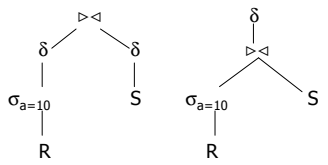
- |R ⋈ S|: ??
- Example:
  - R(a,b): T(R)=1000, V(R,b)=20
  - S(b,c): T(S)=2000, V(S,b)=50, V(S,c)=100
  - U(c,d): T(U)=5000, V(U,c)=500
- Join on multiple attributes where $Y=\{y_1,y_2,...,y_n\}$??

Estimating Other Operators

- Projection
- Union, intersection, difference
  - Usually the average of max and min
- Duplicate elimination and group by
  - $V(R,[a_1,a_2,...a_n])$

Example: Plan Selection

- R(a,b)
  - T(R) = 5000, V(R,a) = 50, V(R,b) = 100
- S(b,c)
  - T(S) = 2000, V(S,b) = 200, V(S,c) = 100



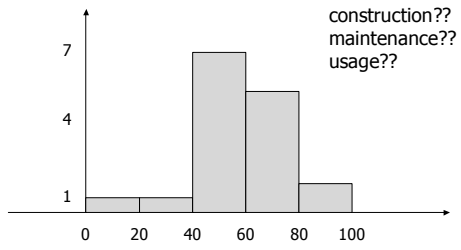
More Statistics

- Criteria
  - Small storage footprint
  - Low computation overhead
  - Accurate estimation
- General techniques
  - Histogram
    - Works very well for low-dimensional data
  - Sampling
    - Works better for high-dimensional data

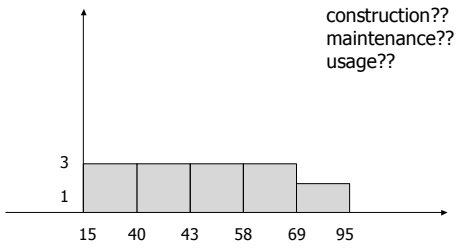
About Histograms

- Construction
  - Sampling
  - buckets
- Maintenance
  - Incremental
  - Periodically re-build
- Usage
  - Uniform assumption

Equi-width Histogram



Equi-depth Histogram



Examples

- Estimate | $\sigma_{score=60}$ |
- Estimate | $\sigma_{score<=60 \text{ and } score>=50}$ |

Join Order

- How many different ways can we join R, S, T??
- How about R_1, R_2, ..., R_n??
- Number of tree shapes:

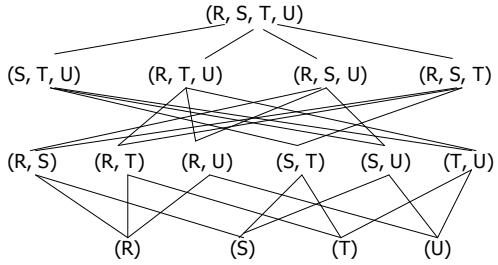
$$T(1) = 1$$

$$T(n) = \sum_{i=1}^{n-1} T(i)T(n-i)$$

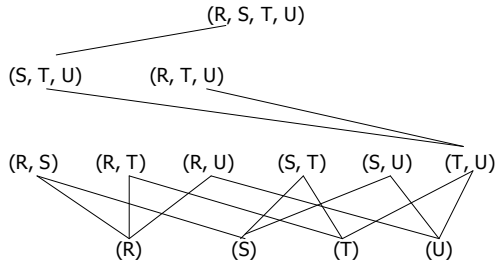
Select Join Order

- Consider only *left-deep* trees: we still have n! choices
  - Dynamic programming
  - Greedy

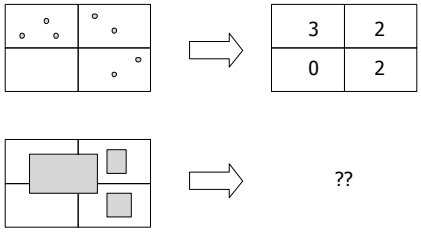
Dynamic Programming



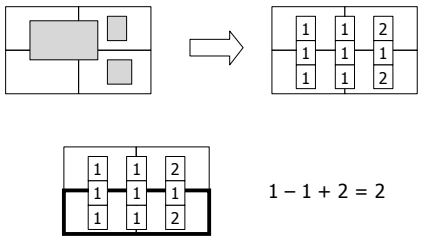
Greedy



Histogram for Spatial Data



Euler Histogram



Physical Query Plans

- Plan enumeration
- Operator selection
- Pipeline vs. Materialization

Readings

- Stanford book: Chapter 16
- [Euler histogram paper]