PURDUE UNIVERSITY

CS54100: Database Systems

Query Processing

19 March 2012

Prof. Chris Clifton

Indiana Center for Database Systems ™

Query Processing

- Q → Query Plan

Focus: Relational System

- Others?

CS54100

Example

Select B,D

From R,S

Where R.A = “c” $\wedge$ S.E = 2 $\wedge$ R.C=S.C

CS54100

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

CS54100

How do we execute query?

One idea

- Do Cartesian product
- Select tuples
- Do projection

RXS

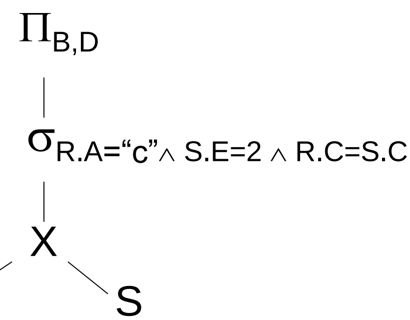
R.A	R.B	R.C	S.C	S.D	S.E
a	1	10	10	x	2
a	1	10	20	y	2
.					
.					
C	2	10	10	x	2
.					
.					

Bingo! →

Got one...

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

Ex: Plan I

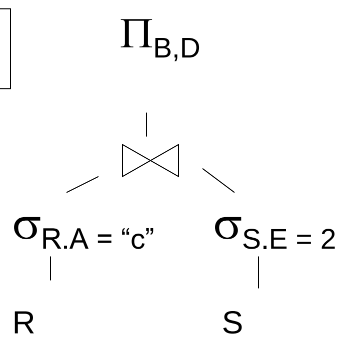


OR: $\Pi_{B,D} [\sigma_{R.A=\text{“c”} \wedge S.E=2 \wedge R.C = S.C} (RXS)]$

CS54100

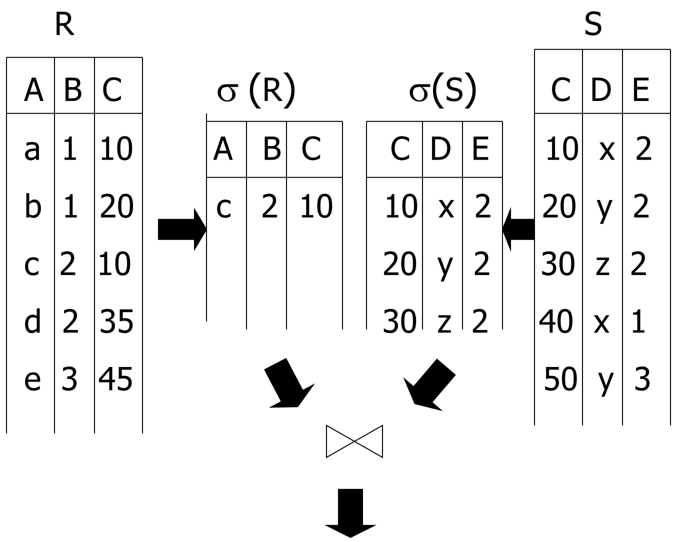
Another idea:

Plan II

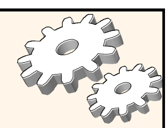


$\bowtie$ natural join

CS54100



CS54100



Overview of Query Evaluation

- ***Plan:*** *Tree of R.A. ops, with choice of alg for each op.*
  - Each operator typically implemented using a \`pull' interface: when an operator is \`pulled' for the next output tuples, it \`pulls' on its inputs and computes them.
- Two main issues in query optimization:
  - For a given query, what plans are considered?
    - Algorithm to search plan space for cheapest (estimated) plan.
  - How is the cost of a plan estimated?
- Ideally: Want to find best plan. Practically: Avoid worst plans!
- We will study the System R approach.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 10

Plan III

Use R.A and S.C Indexes

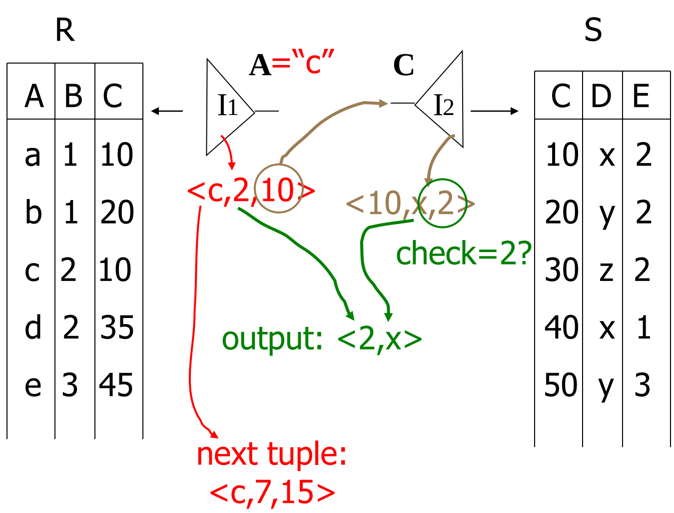
(1) Use R.A index to select R tuples with R.A = “c”

(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

CS54100



CS54100

Some Common Techniques

- Algorithms for evaluating relational operators use some simple ideas extensively:
  - Indexing: Can use WHERE conditions to retrieve small set of tuples (selections, joins)
  - Iteration: Sometimes, faster to scan all tuples even if there is an index. (And sometimes, we can scan the data entries in an index instead of the table itself.)
  - Partitioning: By using sorting or hashing, we can partition the input tuples and replace an expensive operation by similar operations on smaller inputs.

** Watch for these techniques as we discuss query evaluation!*

Access Paths

- An <u>access path</u> is a method of retrieving tuples:
  - File scan, or index that matches a selection (in the query)
- A tree index <u>*matches*</u> (a conjunction of) terms that involve only attributes in a *prefix* of the search key.
  - E.g., Tree index on <*a, b, c*> matches the selection *a*=5 *AND b*=3, and *a*=5 *AND b*>6, but not *b*=3.
- A hash index <u>*matches*</u> (a conjunction of) terms that has a term *attribute = value* for every attribute in the search key of the index.
  - E.g., Hash index on <*a, b, c*> matches *a*=5 *AND b*=3 *AND c*=5; but it does not match *b*=3, *or a*=5 *AND b*=3, *or a*>5 *AND b*=3 *AND c*=5.

A Note on Complex Selections

(day<8/9/94 AND rname='Paul') OR bid=5 OR sid=3

- Selection conditions are first converted to *conjunctive normal form* (CNF):

  (day<8/9/94 OR bid=5 OR sid=3) AND
  (rname='Paul' OR bid=5 OR sid=3)

- We only discuss case with no ORs; see text if you are curious about the general case.

Using an Index for Selections

- Cost depends on #qualifying tuples, and clustering.
  - Cost of finding qualifying data entries (typically small) plus cost of retrieving records (could be large w/o clustering).
  - In example, assuming uniform distribution of names, about 10% of tuples qualify (100 pages, 10000 tuples). With a clustered index, cost is little more than 100 I/Os; if unclustered, upto 10000 I/Os!

```
SELECT  *
FROM    Reserves R
WHERE   R.rname < 'C%'
```


One Approach to Selections

- Find the *most selective access path*, retrieve tuples using it, and apply any remaining terms that don't match the index:
  - *Most selective access path:* An index or file scan that we estimate will require the fewest page I/Os.
  - Terms that match this index reduce the number of tuples *retrieved*; other terms are used to discard some retrieved tuples, but do not affect number of tuples/pages fetched.
  - Consider *day<8/9/94 AND bid=5 AND sid=3*. A B+ tree index on *day* can be used; then, *bid=5* and *sid=3* must be checked for each retrieved tuple. Similarly, a hash index on <*bid, sid*> could be used; *day<8/9/94* must then be checked.

Projection

```
SELECT  DISTINCT
        R.sid, R.bid
FROM    Reserves R
```

- The expensive part is removing duplicates.
  - SQL systems don't remove duplicates unless the keyword DISTINCT is specified in a query.
- Sorting Approach: Sort on <sid, bid> and remove duplicates. (Can optimize this by dropping unwanted information while sorting.)
- Hashing Approach: Hash on <sid, bid> to create partitions. Load partitions into memory one at a time, build in-memory hash structure, and eliminate duplicates.
- If there is an index with both R.sid and R.bid in the search key, may be cheaper to sort data entries!

Overview of Query Optimization

CS54100

PURDUE UNIVERSITY

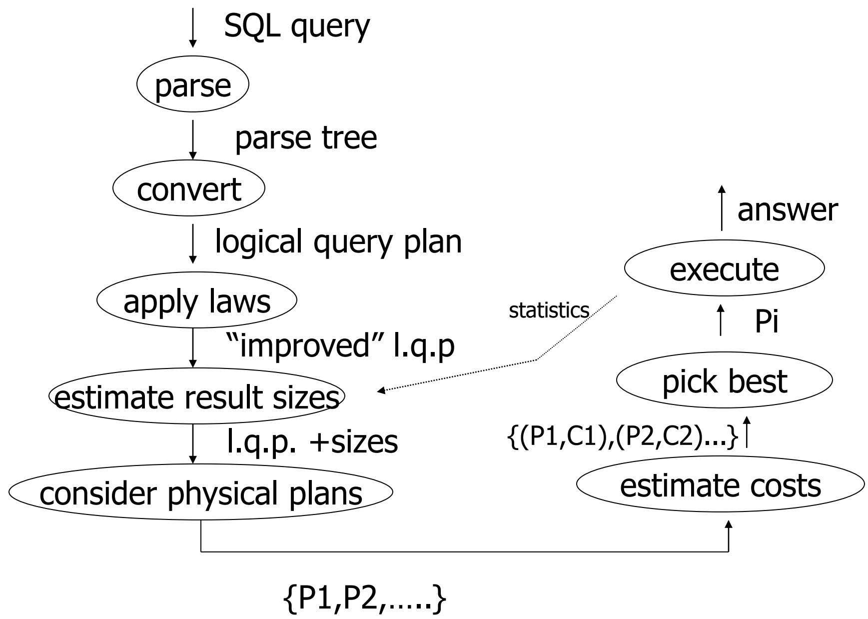
CS54100: Database Systems

Query Processing

21 March 2012

Prof. Chris Clifton

Indiana Center for Database Systems ™



CS54100

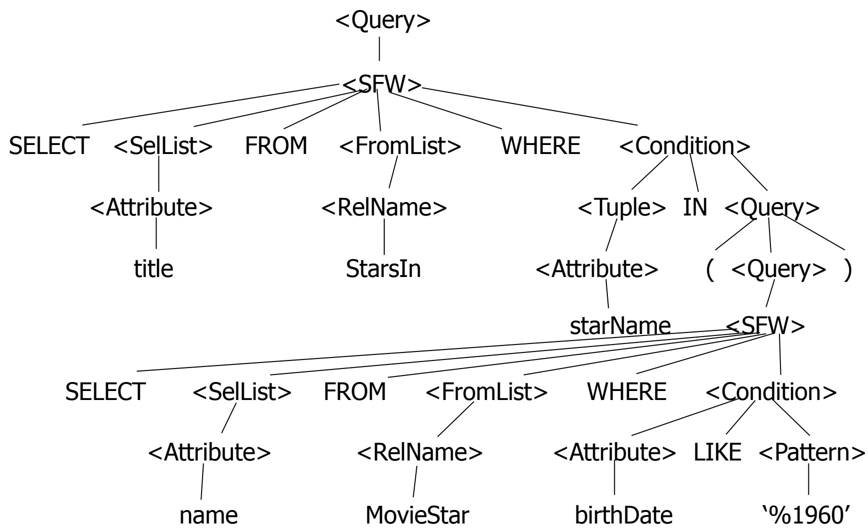
Example: SQL query

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

(Find the movies with stars born in 1960)

CS54100

Example: Parse Tree



CS54100

Example: Generating Relational Algebra

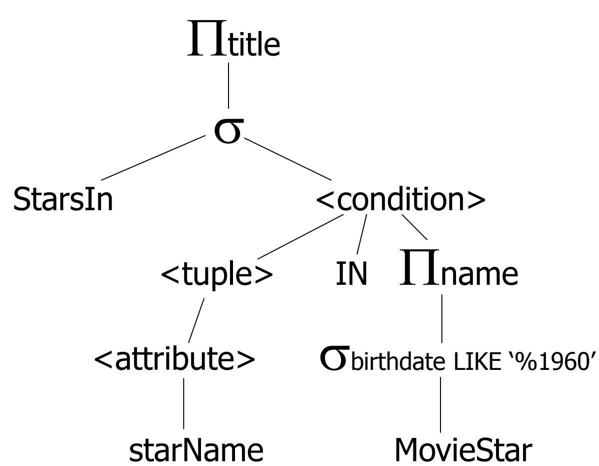


Fig. 7.15: An expression using a two-argument σ, midway between a parse tree and relational algebra

CS54100

Example: Logical Query Plan

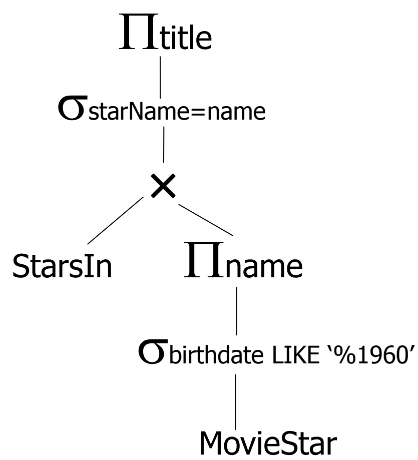
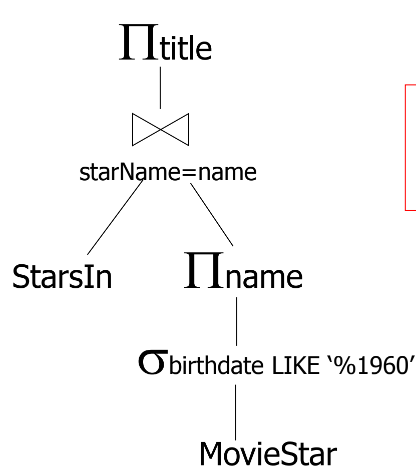


Fig. 7.18: Applying the rule for IN conditions

CS54100

Example: Improved Logical Query Plan

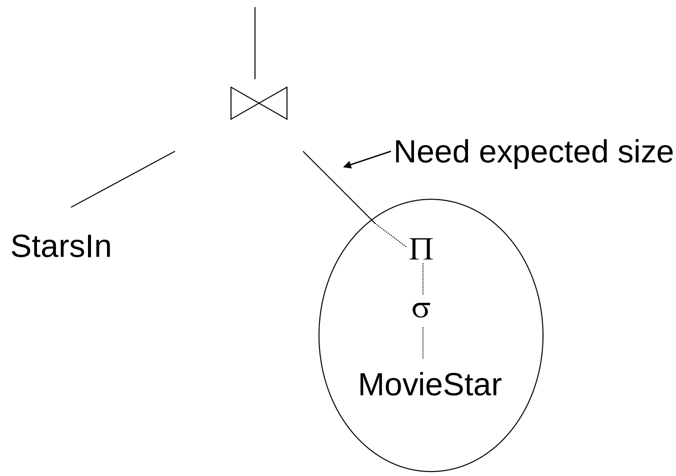


Question: Push project to StarsIn?

Fig. 7.20: An improvement on fig. 7.18.

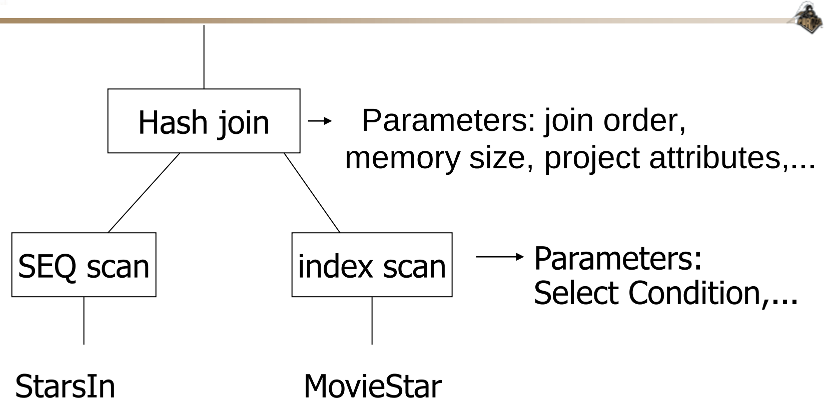
CS54100

Example: Estimate Result Sizes



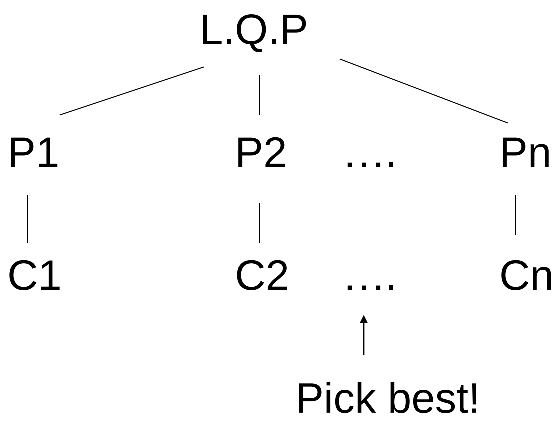
CS54100

Example: One Physical Plan



CS54100

Example: Estimate costs



Join: Index Nested Loops

foreach tuple r in R do
  foreach tuple s in S where $r_i == s_j$ do
    add <r, s> to result

- If there is an index on the join column of one relation (say S), can make it the inner and exploit the index.
  - Cost: M + (($M*p_R$) * cost of finding matching S tuples)
  - M=#pages of R, p_R=# R tuples per page
- For each R tuple, cost of probing S index is about 1.2 for hash index, 2-4 for B+ tree. Cost of then finding S tuples (assuming Alt. (2) or (3) for data entries) depends on clustering.
  - Clustered index: 1 I/O (typical), unclustered: upto 1 I/O per matching S tuple.

Examples of Index Nested Loops

- Hash-index (Alt. 2) on *sid* of Sailors (as inner):
  - Scan Reserves: 1000 page I/Os, 100*1000 tuples.
  - For each Reserves tuple: 1.2 I/Os to get data entry in index, plus 1 I/O to get (the exactly one) matching Sailors tuple. Total: 220,000 I/Os.
- Hash-index (Alt. 2) on *sid* of Reserves (as inner):
  - Scan Sailors: 500 page I/Os, 80*500 tuples.
  - For each Sailors tuple: 1.2 I/Os to find index page with data entries, plus cost of retrieving matching Reserves tuples. Assuming uniform distribution, 2.5 reservations per sailor (100,000 / 40,000). Cost of retrieving them is 1 or 2.5 I/Os depending on whether the index is clustered.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 31

Join: Sort-Merge ($R \bowtie_{i=j} S$)

- Sort R and S on the join column, then scan them to do a ``merge'' (on join col.), and output result tuples.
  - Advance scan of R until current R-tuple >= current S tuple, then advance scan of S until current S-tuple >= current R tuple; do this until current R tuple = current S tuple.
  - At this point, all R tuples with same value in Ri (*current R group*) and all S tuples with same value in Sj (*current S group*) *match*; output <r, s> for all pairs of such tuples.
  - Then resume scanning R and S.
- R is scanned once; each S group is scanned once per matching R tuple. (Multiple scans of an S group are likely to find needed pages in buffer.)

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 32

Example of Sort-Merge Join

sid	sname	rating	age
22	dustin	7	45.0
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0

sid	bid	day	rname
28	103	12/4/96	guppy
28	103	11/3/96	yuppy
31	101	10/10/96	dustin
31	102	10/12/96	lubber
31	101	10/11/96	lubber
58	103	11/12/96	dustin

- Cost: M log M + N log N + (M+N)
  - The cost of scanning, M+N, could be M*N (very unlikely!)
- With 35, 100 or 300 buffer pages, both Reserves and Sailors can be sorted in 2 passes; total join cost: 7500.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 33

Query Optimization - In class order

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

CS54100

Relational algebra optimization

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

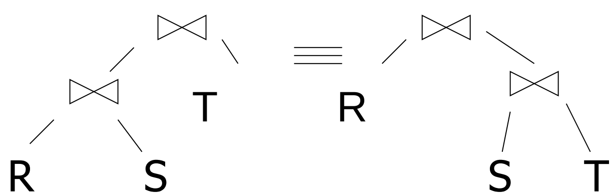
Rules: Natural joins & cross products & union

$R \bowtie S = S \bowtie R$

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

Note:

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



Rules: Natural joins & cross products & union

$R \bowtie S = S \bowtie R$

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

- 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

Rules: Selects

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

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

CS54100

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}

CS54100

Option 2 (MAX) makes this rule work:

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

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

P1 satisfied by a,b; P2 satisfied by b,c

$\sigma_{p1 \vee p2}(R)$ = {a,a,b,b,b,c}

$\sigma_{p1}(R)$ = {a,a,b,b,b}

$\sigma_{p2}(R)$ = {b,b,b,c}

$\sigma_{p1}(R) \cup \sigma_{p2}(R)$ = {a,a,b,b,b,c}

CS54100

"Sum" option makes more sense:

Senators (……) Rep (……)

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

T1

Yr	State
97	CA
99	CA
98	AZ

T2

Yr	State
99	CA
99	CA
98	CA

Union?

CS54100

Executive Decision

-> Use "SUM" option for bag unions

-> Some rules cannot be used for bags

CS54100

PURDUE UNIVERSITY

CS54100: Database Systems

Query Processing

23 March 2012

Prof. Chris Clifton

Indiana Center for Database Systems ™

Rules: Project

Let: X = set of attributes

Y = set of attributes

XY = X U Y

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

CS54100

Rules: $\sigma + \bowtie$ combined

Let p = predicate with only R attribs

q = predicate with only S attribs

m = predicate with only R,S attribs

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

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

CS54100

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

Some Rules can be Derived:

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

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

$\sigma_{p \vee q} (R \bowtie S) =$

CS54100

--> Derivation for first one:

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

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

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

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

CS54100

Rules: π,σ combined

Let x = subset of R attributes

z = attributes in predicate P

(subset of R attributes)

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

CS54100

Rules: $\pi, \bowtie$ combined

Let x = subset of R attributes

y = subset of S attributes

z = intersection of R,S attributes

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

$$\pi_{xy}\{[\pi_{xz}(R)] \bowtie [\pi_{yz}(S)]\}$$

CS54100

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

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

z' = z U {attributes used in P }

CS54100

Rules for σ, π combined with X

similar...

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

CS54100

Rules σ, U combined:

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

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

CS54100

Which are "good" transformations?

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

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

$R \bowtie S \rightarrow S \bowtie R$

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

CS54100

Conventional wisdom: do projects early

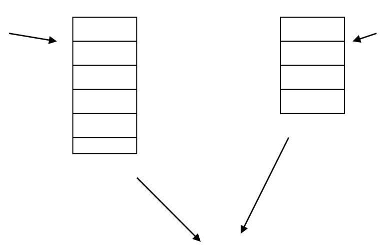
Example: R(A,B,C,D,E) x={E}
P: (A=3) $\wedge$ (B="cat")

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

What if we have A, B indexes?

B = "cat"

A=3



Intersect pointers to get pointers to matching tuples

Bottom line:

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

CS54100

More transformations

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

CS54100

Outline - Query Processing

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

CS54100

Estimating cost of query plan

1. Estimating size of results
2. Estimating # of IOs

CS54100

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

CS54100

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

CS54100

Size estimates for W = R1 x R2

T(W) = T(R1) × T(R2)

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

CS54100

Size estimate for W = s A=a (R)

- S(W) = S(R)
- T(W) = ?

CS54100

CS54100: Database Systems

Query Processing

26 March 2012

Prof. Chris Clifton

Indiana Center for Database Systems ™

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

V(R,A)=3

V(R,B)=1

V(R,C)=5

V(R,D)=4

$$W = \sigma_{z=val}(R) \quad T(W) = \frac{T(R)}{V(R,Z)}$$

CS54100

Assumption:

Values in select expression Z = val
are uniformly distributed
over possible V(R,Z) values.

Alternate Assumption:

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

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

Alternate assumption
V(R,A)=3 DOM(R,A)=10
V(R,B)=1 DOM(R,B)=10
V(R,C)=5 DOM(R,C)=10
V(R,D)=4 DOM(R,D)=10

$W = \sigma_{z=val}(R)$ $T(W) = ?$

CS54100

C=val $\Rightarrow$ $T(W) = (1/10)1 + (1/10)1 + \ldots$
$= (5/10) = 0.5$

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

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

CS54100

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

Alternate assumption
V(R,A)=3 DOM(R,A)=10
V(R,B)=1 DOM(R,B)=10
V(R,C)=5 DOM(R,C)=10
V(R,D)=4 DOM(R,D)=10

$$W = \sigma_{z=val}(R) \quad T(W) = \frac{T(R)}{DOM(R,Z)}$$

CS54100

Selection cardinality

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

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

CS54100

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

- Solution # 1:

  T(W) = T(R)/2

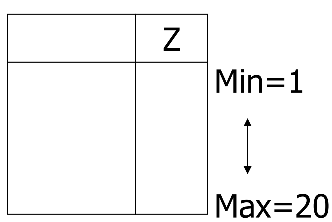
- Solution # 2:

  T(W) = T(R)/3

CS54100

- Solution # 3: Estimate values in range

<u>Example</u> R



V(R,Z)=10

$W = \sigma_{z \geq 15}(R)$

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

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

CS54100

Equivalently:

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

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

CS54100

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

CS54100

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

R1

A	B	C

R2

A	D

Assumption:

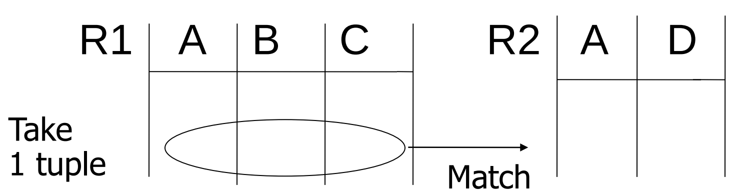
$V(R1,A) \leq V(R2,A) \Rightarrow$ Every A value in R1 is in R2

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

“containment of value sets” Sec. 7.4.4

CS54100

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



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

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

CS54100

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

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

[A is common attribute]

CS54100

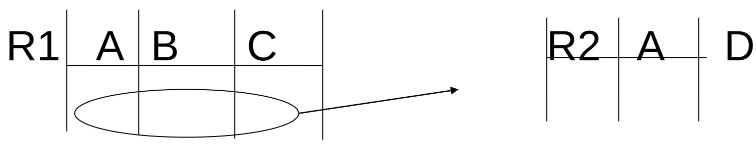
In general W = R1 ⋈ R2

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

CS54100

Case 2 with alternate assumption

Values uniformly distributed over domain



This tuple matches T(R2)/DOM(R2,A) so

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

Assume the same

CS54100

In all cases:

$$S(W) = S(R1) + S(R2) - S(A)$$

size of attribute A

CS54100

Using similar ideas, we can estimate sizes of:

$\Pi_{AB}(R)$ ….. Sec. 16.4.2

$\sigma_{A=a \wedge B=b}(R)$ …. Sec. 16.4.3

$R \bowtie S$ with common attribs. A,B,C
Sec. 16.4.5

Union, intersection, diff, …. Sec. 16.4.7

CS54100

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

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

$[\sigma_{A=a}(R1)]$: Treat as relation U

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

Also need V (U, *) !!

CS54100

To estimate Vs

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

Say R1 has attribs A,B,C,D

V(U, A) =

V(U, B) =

V(U, C) =

V(U, D) =

Example

R1

A	B	C	D
cat	1	10	10
cat	1	20	20
dog	1	30	10
dog	1	40	30
bat	1	50	10

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)

CS54100

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)

CS54100

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

CS54100

Partial Result: $U = R \bowtie S$

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

V(U,A) = 50

V(U,B) = 100

V(U,C) = 300

CS54100

$Z = U \bowtie R3$

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

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

CS54100

Summary

- Estimating size of results is still a mix of science and art
  - *Research opportunity?*
- But it does work reasonably well
- Don't forget:

  Statistics must be kept up to date…

  (cost?)

CS54100

Outline

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

CS54100