Database System Principles

Query Processing

Query Processing

Q → Query Plan

Focus: Relational System

- Others? OODBMS, MMDBMS?

Example

Select B,D

From R,S

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

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

- 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
.					
.					

Got one...

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

Ex: Plan I

$\Pi_{B,D}$ — Do projection

$\sigma_{R.A=\text{“}c\text{”} \wedge S.E=2 \wedge R.C=S.C}$ — Select tuples

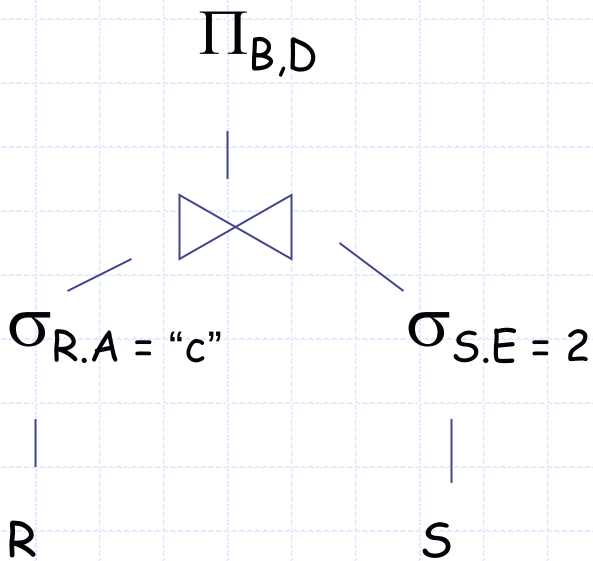
$\times$ — Do Cartesian product

R S

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

Another idea:

Plan II



$\bowtie$ natural join

R

A	B	C
a	1	10
b	1	20
c	2	10
d	2	35
e	3	45

R.A = “c”

σ (R)

A	B	C
c	2	10

S.E = 2

σ(S)

C	D	E
10	x	2
20	y	2
30	z	2

S

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

$\bowtie$

Plan III

Use R.A and S.C Indexes

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

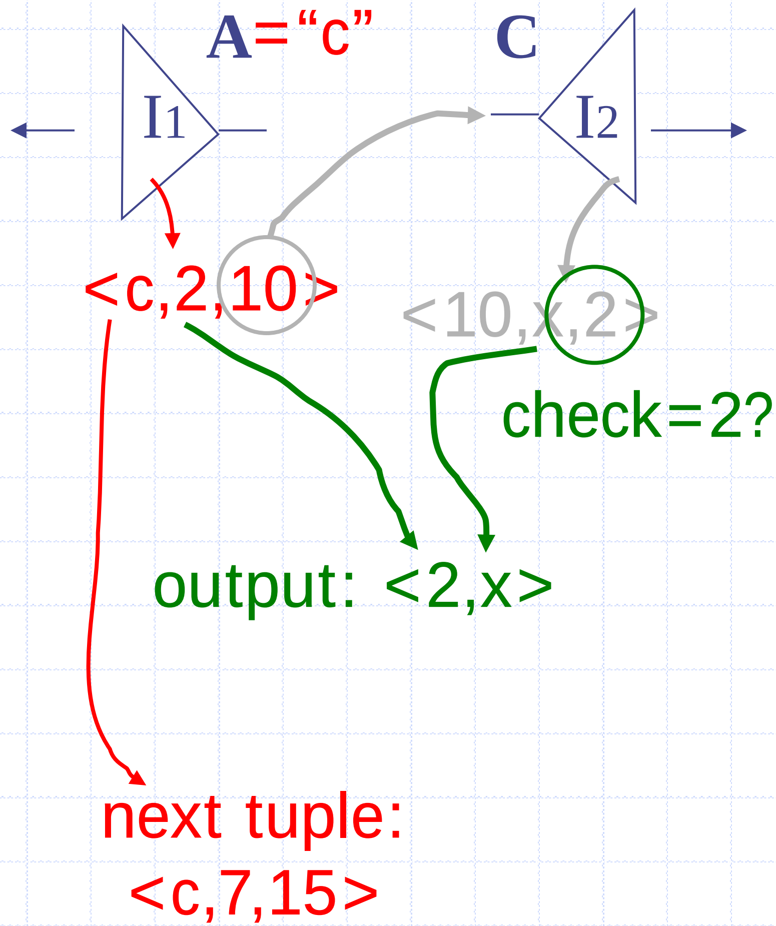
(2) For each R.C value found, use S.C index to find matching tuples

(3) Eliminate S tuples: S.E $\neq$ 2

(4) Join matching R,S tuples, project B,D attributes and place in result

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

Overview of Query Optimization

查询优化的途径：

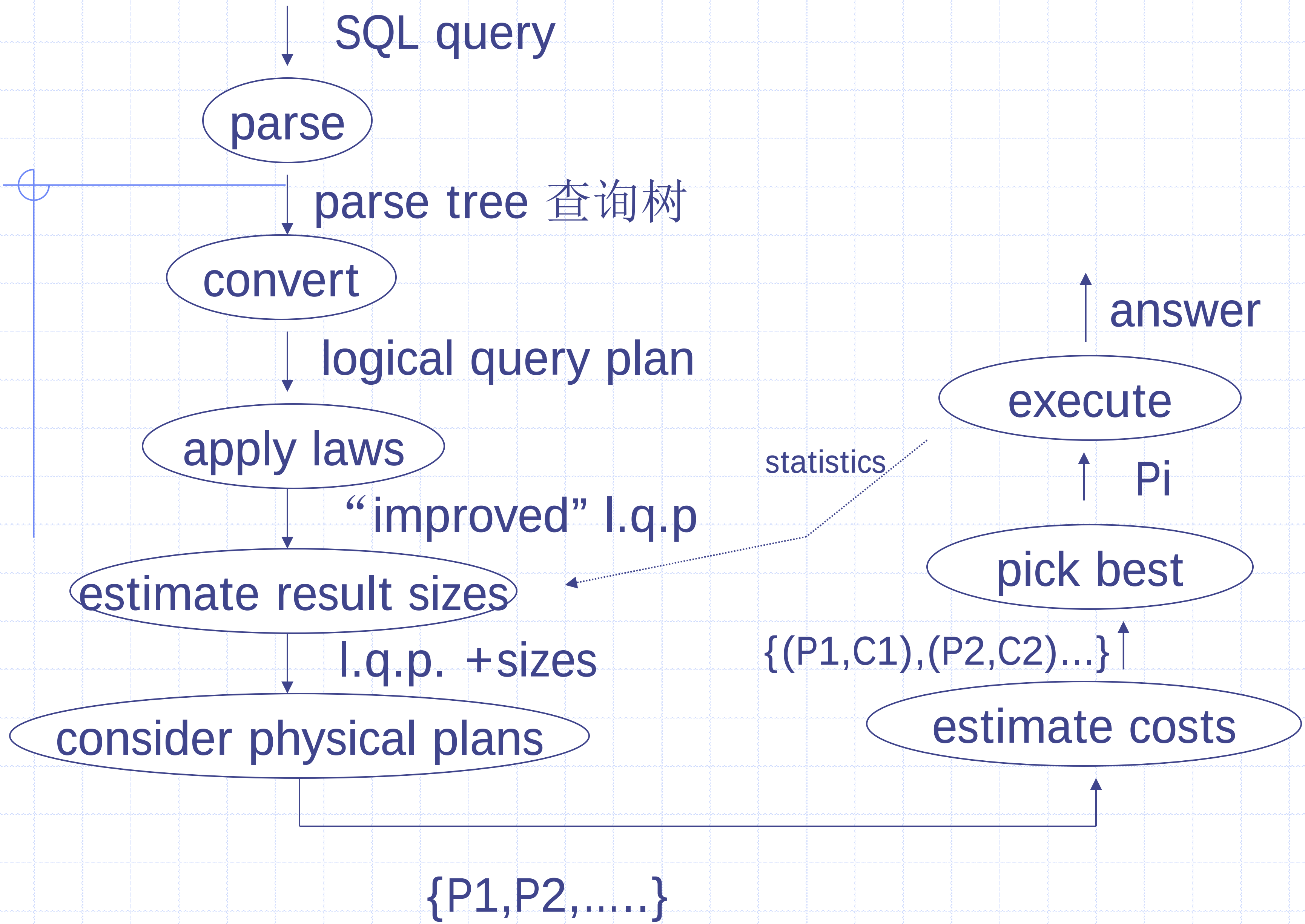
1代数优化：对查询语句进行变换，改变操作的次序，使查询更有效。

2物理优化：根据系统提供的存取路径，选择合理的存取策略，例如顺序搜索或者索引进行查询，这种依赖于物理存取路径的优化，称为物理优化。

3规则优化：根据启发式规则，选择执行的策略，如先做选择、投影等一元操作，后做连接操作等。

4代价估算：对可供选择的执行策略进行代价估算，从中选择代价最小的执行策略。

数据库系统往往综合运用上述优化方法


SQL query
parse
parse tree 查询树
convert
logical query plan
apply laws
“improved” l.q.p
estimate result sizes
l.q.p. +sizes
consider physical plans
{P1,P2,.....}
estimate costs
{(P1,C1),(P2,C2)...}
pick best
Pi
execute
answer
statistics

Example: SQL query

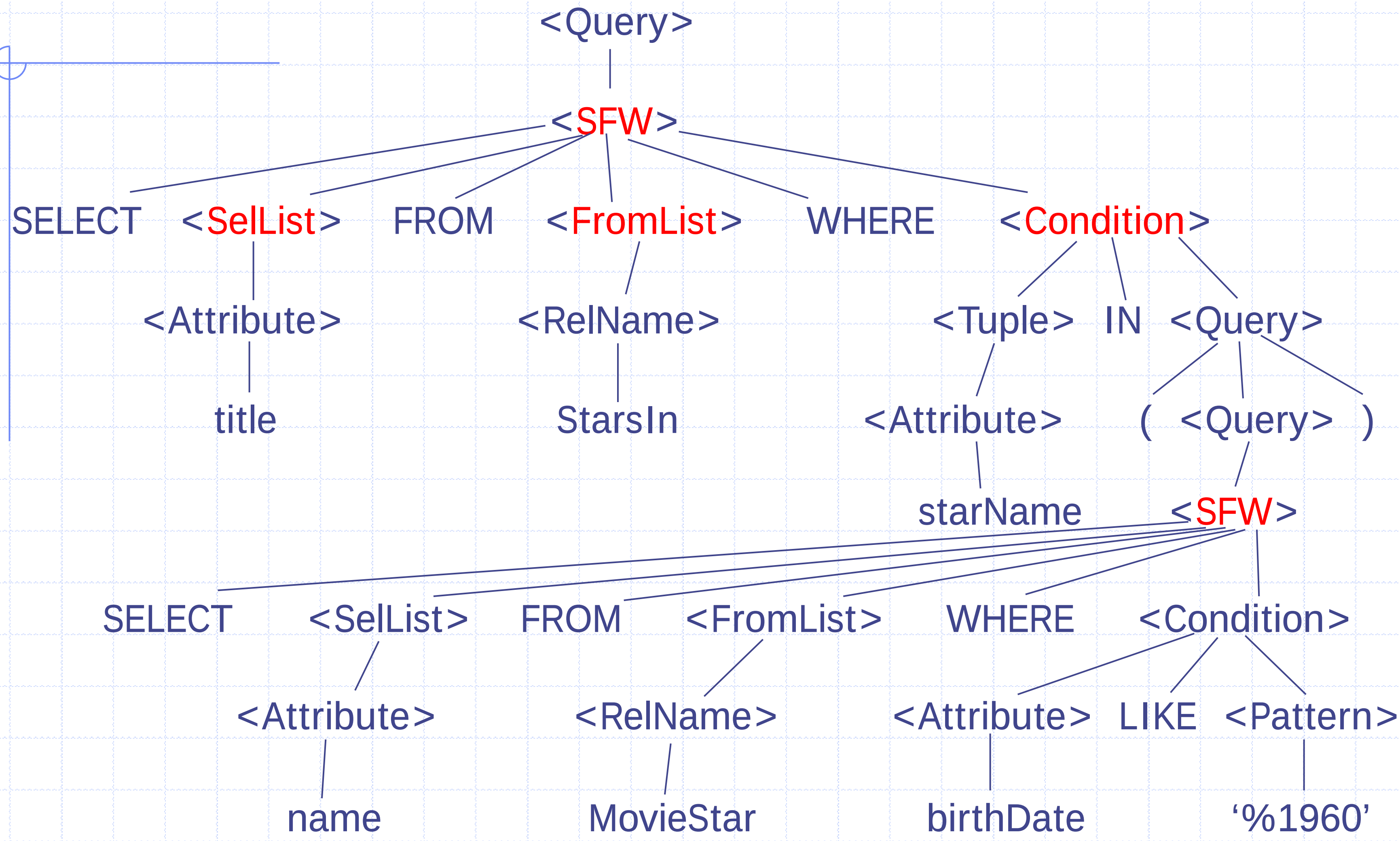
Find the movies with stars born in 1960

MovieStar (name, birthdate, …)

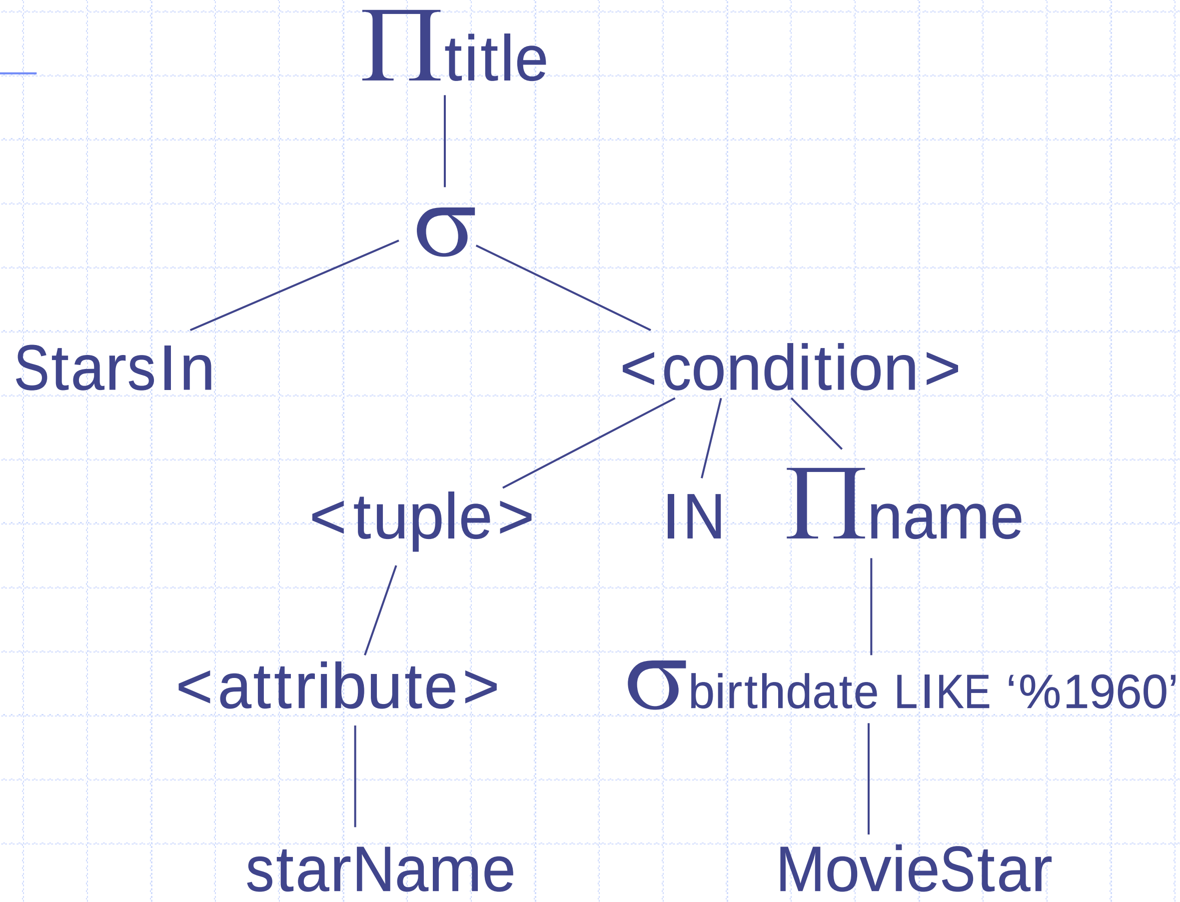
StarsIn (title, starName, …)

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

Example: Parse Tree



Example: Generating Relational Algebra



Example: Logical Query Plan

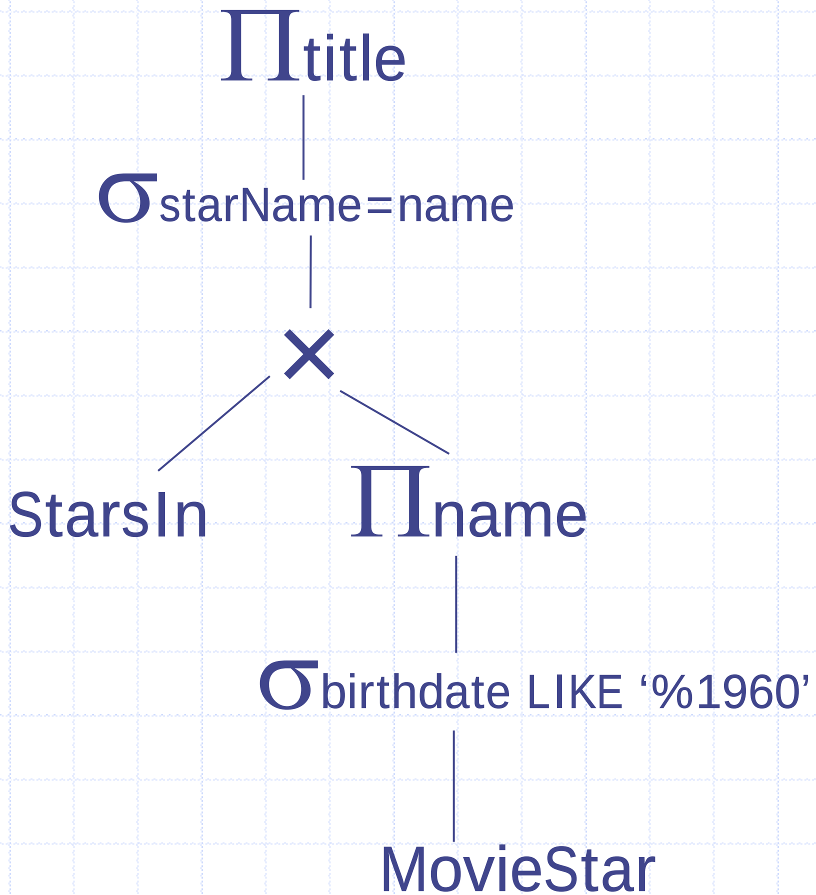


Fig. 7.18: Applying the rule for IN conditions

Example: Improved Logical Query Plan

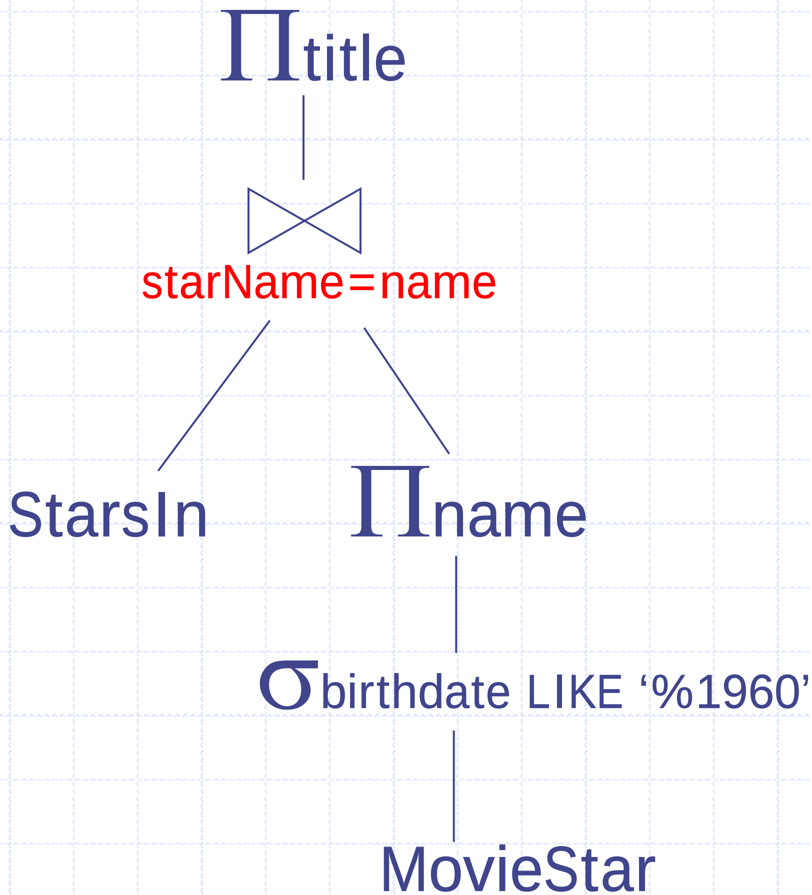
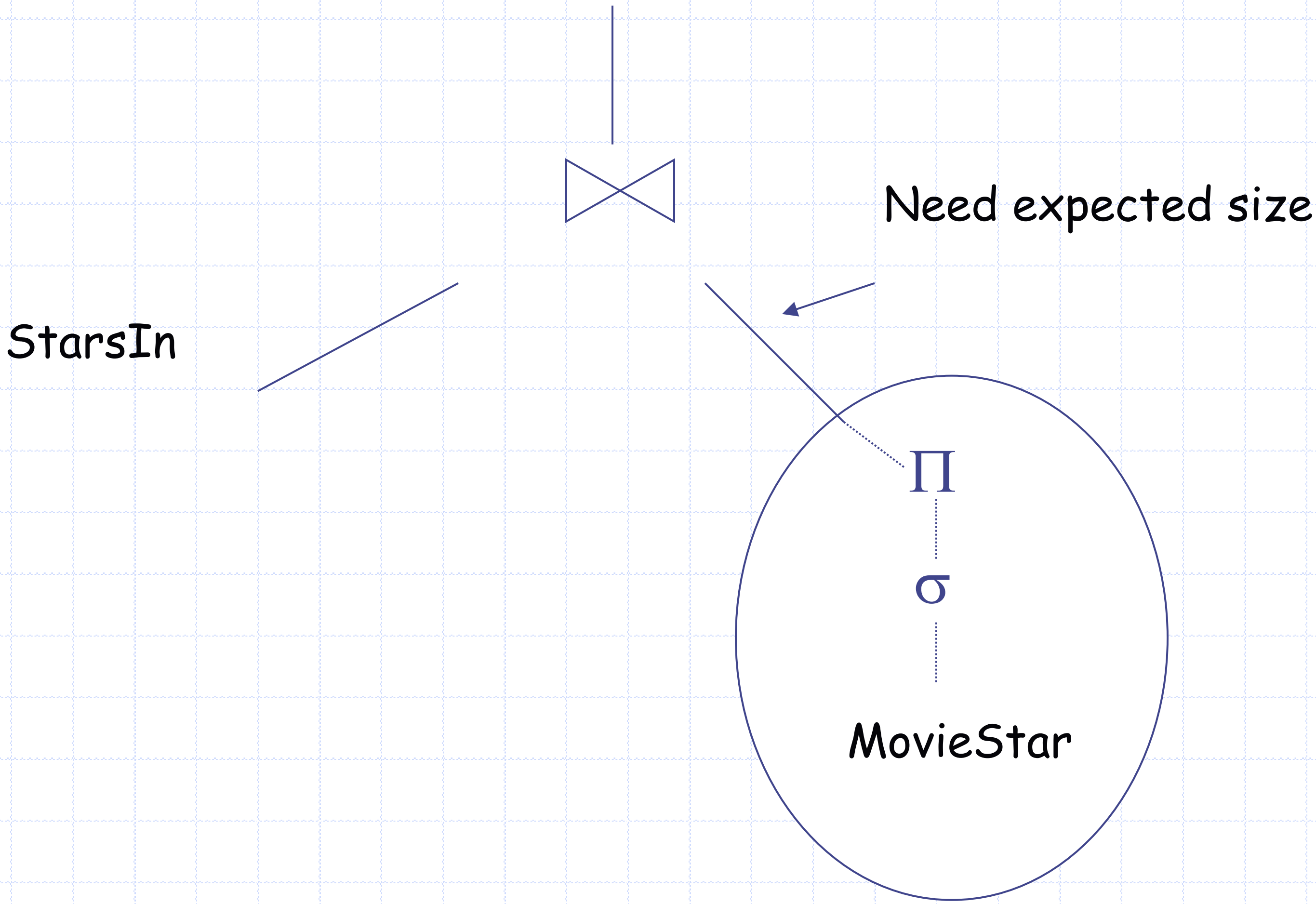
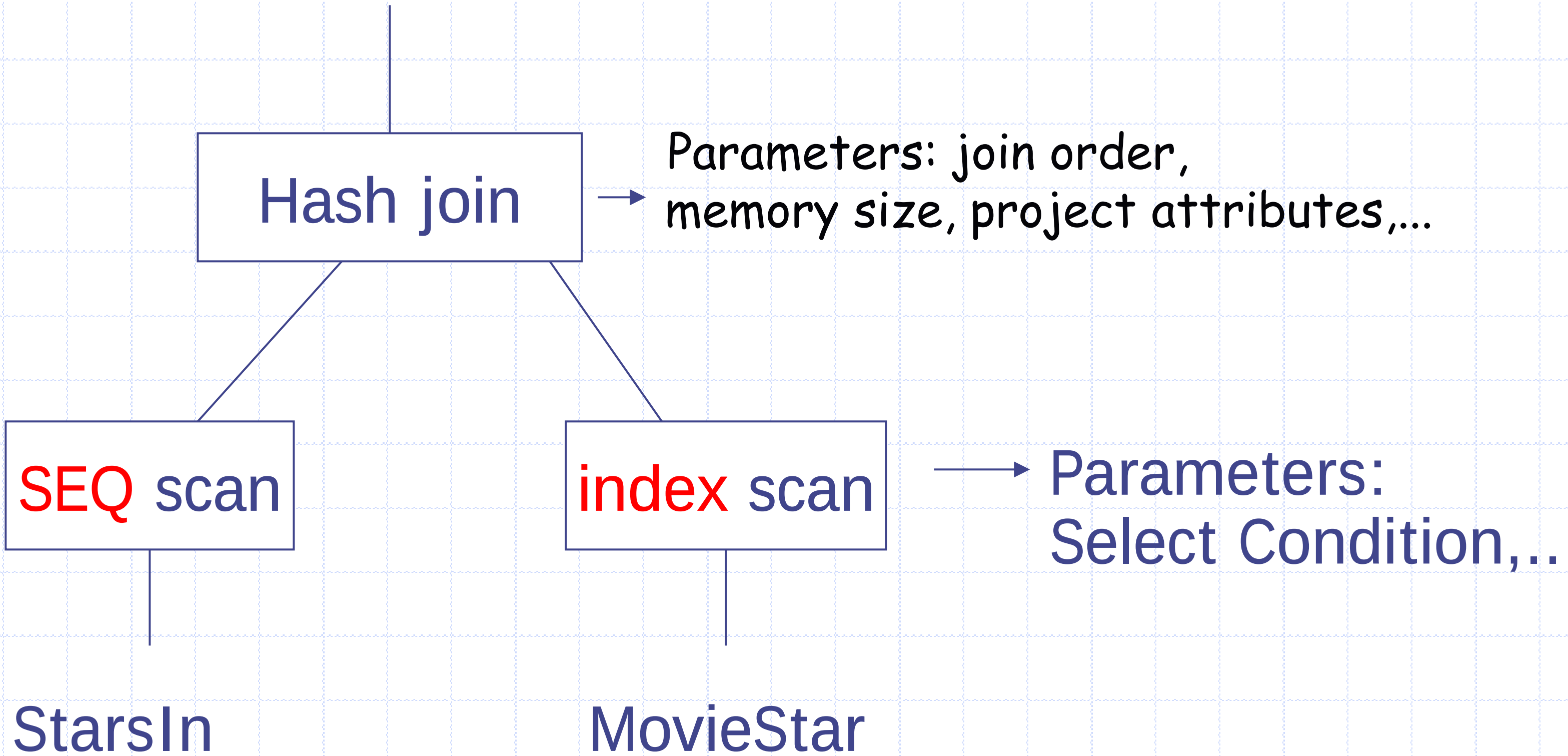


Fig. 7.20: An improvement on fig. 7.18.

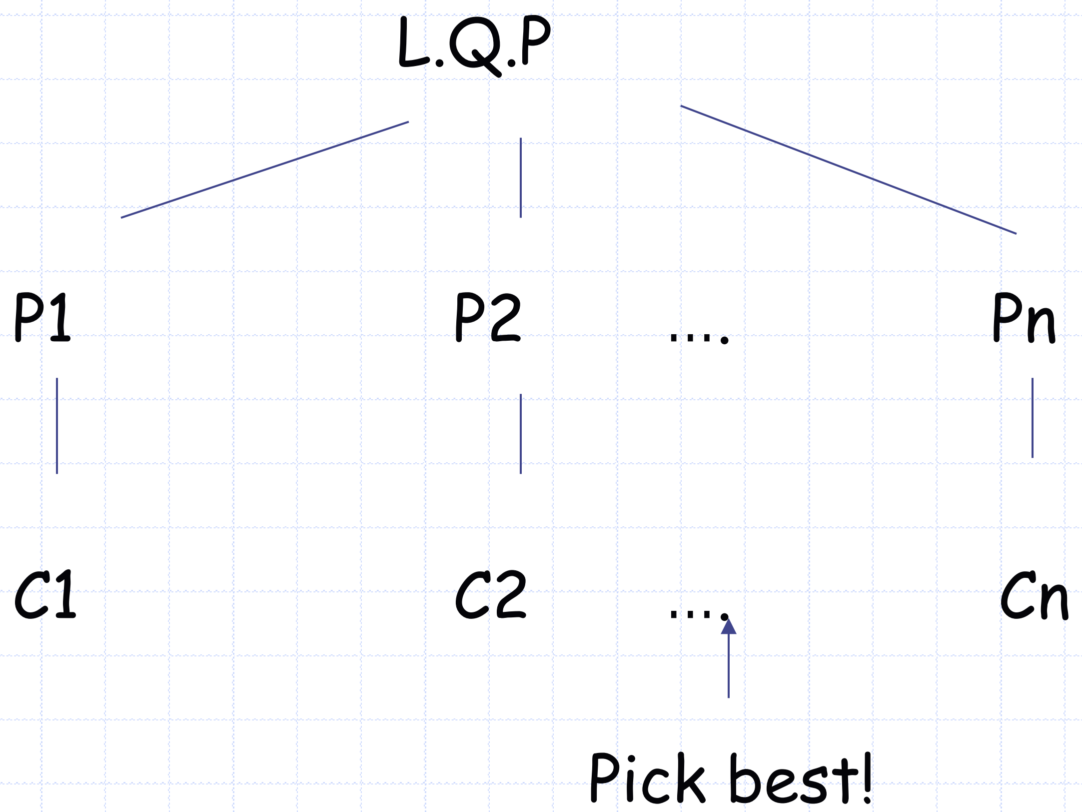
Example: Estimate Result Sizes



Example: One Physical Plan



Example: Estimate costs



Textbook outline

Chapter 6

6.1 Algebra for queries [bags vs sets]

- Select, project, join, ….
- Duplicate elimination

6.2 Physical operators

- Scan,sort, …

6.3-6.10 Implementing operators + estimating their cost

Chapter 7

7.1 Parsing

7.2 Algebraic laws

7.3 Parse tree -> logical query plan

7.4 Estimating result sizes

7.5-7.7 Cost based optimization

Reading textbook - Chapters 6,7

Optional: 6.8, 6.9, 6.10, 7.6, 7.7

Optional: Duplicate elimination operator
grouping, aggregation operators

Next: Query Optimization

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 (chapter 7.2)

- Transformation rules变换规则 (保证等价性 equivalence)
- What are good transformations?

Rules: Natural joins & cross products & union

$R \bowtie S = S \bowtie R$ 交换律

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

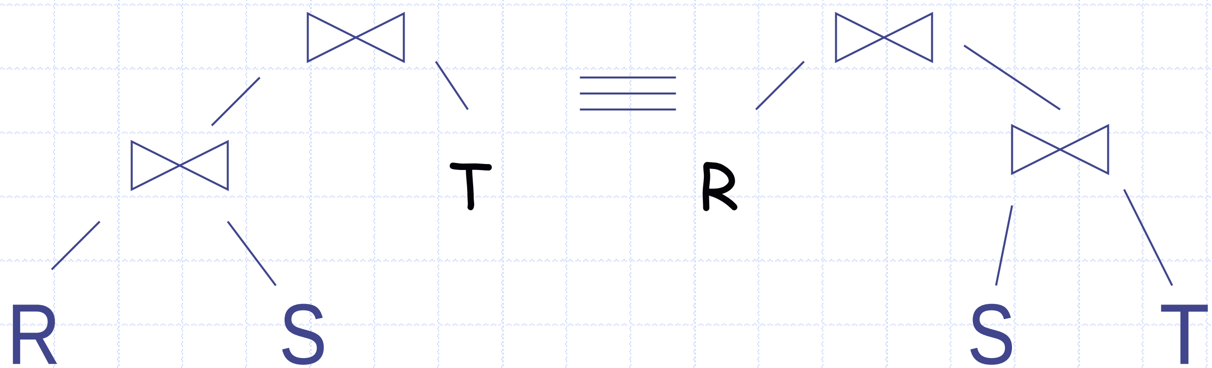
Natural joins： contain all attributes of R and S, except that one copy of each pair of equated attributes is omitted.

cross products： R×S, consists of the attributes of R and the attributes of S.

Union: R U S.

Note:

- Prove by yourself
- Can also write as trees, e.g.:



Rules: Natural joins & cross products & union

$R \bowtie S = S \bowtie R$

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

$R \times S = S \times R$

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

$R \cup S = S \cup R$

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

Rules: Selects

$$\sigma_{p1 \text{ and } p2}(R) = \sigma_{p1}\,[\,\sigma_{p2}(R)]$$

$$\sigma_{p1 \text{ or } p2}(R) = [\,\sigma_{p1}(R)] \cup [\,\sigma_{p2}(R)]$$

Notes:
The second law works only if the Relation R is a set.
If R were a bag, the set-union would have the effect of eliminating duplicates incorrectly.

Executive Decision

-> Some rules cannot be used for bags

SQL: distinct 操作消除重复项

Rules: Project

Let: X = set of attributes

Y = set of attributes

XY = X U Y

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

Rules: $\sigma + \bowtie$ combined

Let p = predicate with only R attributes

q = predicate with only S attributes

m = predicate with only R,S attributes

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

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

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

Some Rules can be Derived:

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

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

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

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

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

$$\sigma_m \left[(\sigma_p R) \bowtie (\sigma_q S) \right]$$

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

$$\left[(\sigma_p R) \bowtie S \right] \cup \left[R \bowtie (\sigma_q S) \right]$$

Rules: π,σ combined

Let x = subset of R attributes

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

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

Rules: π, σ combined

Let x = subset of R attributes

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

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

Rules: π, $\bowtie$ 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 σ, π combined with X

similar...

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

Rules $\sigma, \cup$ 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?

- $\sigma_{p1 \wedge p2}(R) \rightarrow \sigma_{p1}[\sigma_{p2}(R)]$
- $\sigma_{p}(R \bowtie S) \rightarrow [\sigma_{p}(R)] \bowtie S$
- $R \bowtie S \rightarrow S \bowtie R$
- $\pi_{x}[\sigma_{p}(R)] \rightarrow \pi_{x}\{\sigma_{p}[\pi_{xz}(R)]\}$

Conventional wisdom:

do projects early

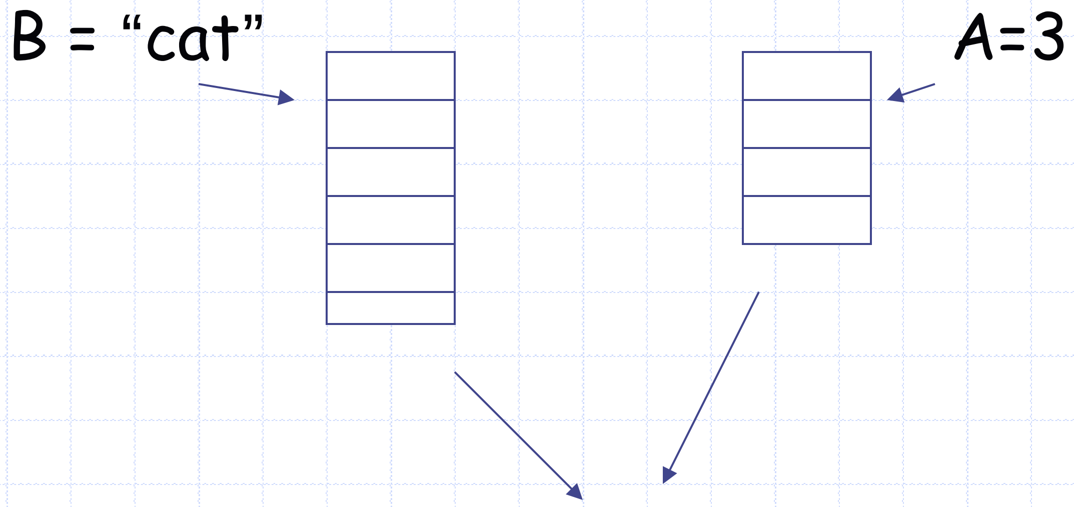
Example: R(A,B,C,D,E) $x=\{E\}$

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

$\pi_x\{\sigma_p\ (R)\}$

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

But What if we have A, B indexes?



Intersect pointers to get pointers to matching tuples

Bottom line:

- Usually good: early selections

In textbook: more transformations

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

小结

- 查询优化的途径：
- √ 1代数优化：对查询语句进行变换，改变操作的次序，使查询更有效。
- 2物理优化：根据系统提供的存取路径，选择合理的存取策略，例如顺序搜索或者索引进行查询，这种依赖于物理存取路径的优化，称为物理优化。
- √ 3规则优化：根据启发式规则，选择执行的策略，如先做选择、投影等一元操作，后做连接操作等。
- 4代价估算：对可供选择的执行策略进行代价估算，从中选择代价最小的执行策略。

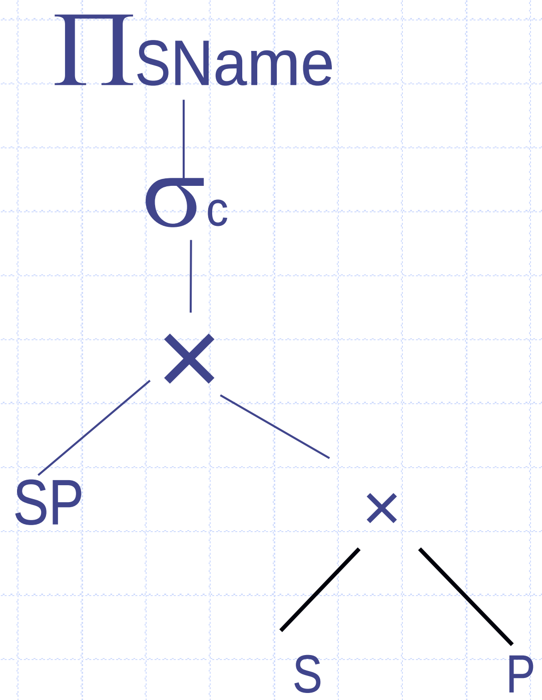
基于规则的代数优化过程：

- 1 以select字句对应投影操作，from字句对应笛卡尔乘积，where字句对应选择条件操作，生成查询计划树；
- 2 应用变换规则，尽可能将选择条件移向树叶的方向；
- 3 应用连接、笛卡尔乘积的结合率，按照小关系先做的原则，重新安排连接的顺序；
- 4 如果笛卡尔乘积还须按连接条件进行选择操作，可将两者组合成连接操作。
- 5 对每个叶结点加必要的投影操作，以消除对查询无用的属性。

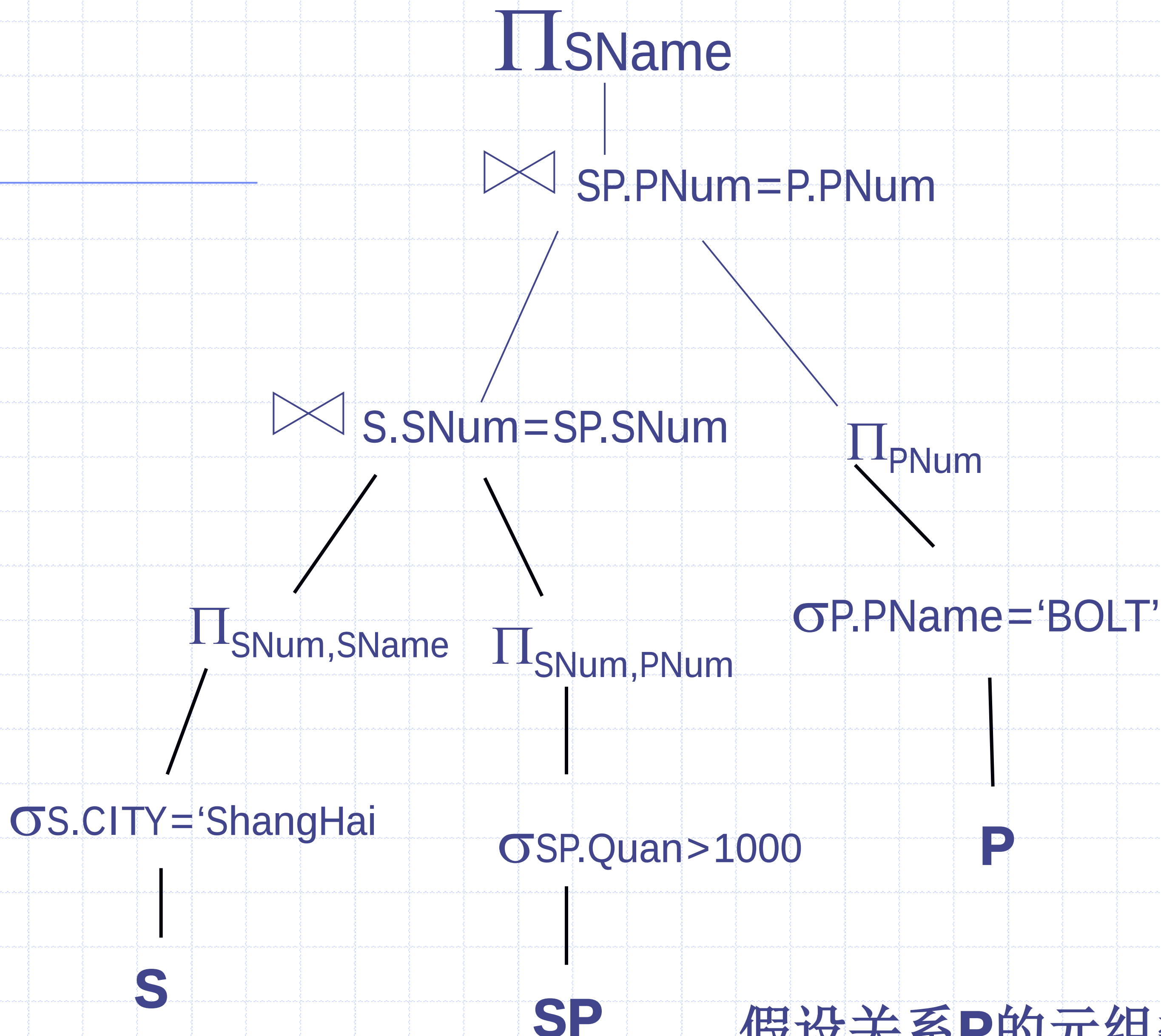
Example

- Given Relation Supplier: S(SNum, SName,City);
- Relation Part: P(PNum, PName,Weight, Size);
- Relation S-P: SP(SNum, PNum,Dept,Quan);
- Then: select SName
- from S,P,SP
- where S.SNum = SP.SNum
- and SP.PNum = P.PNum
- and S.City = 'ShangHai'
- and P.PName = 'Bolt'
- and SP.Quan >1000

Logical Plan Tree



C:查询条件- where 子句;



假设关系P的元组数最多

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

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

$T(R) = 5$ $\qquad S(R) = 37$

$V(R,A) = 3$ $\qquad V(R,C) = 5$

$V(R,B) = 1$ $\qquad V(R,D) = 4$

Size estimates for $W = R1 \times R2$

\# tuples: $T(W) = T(R1) \times T(R2)$

\# of bytes in each tuple

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

Size estimate for $W = \sigma_{A=a}(R)$

$S(W) = S(R)$

$T(W) = ?$

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

$V(R,A)=3$
$V(R,B)=1$
$V(R,C)=5$
$V(R,D)=4$

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

Assumption:

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

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

T(W) = ?

- Solution # 1:

  T(W) = T(R)/2

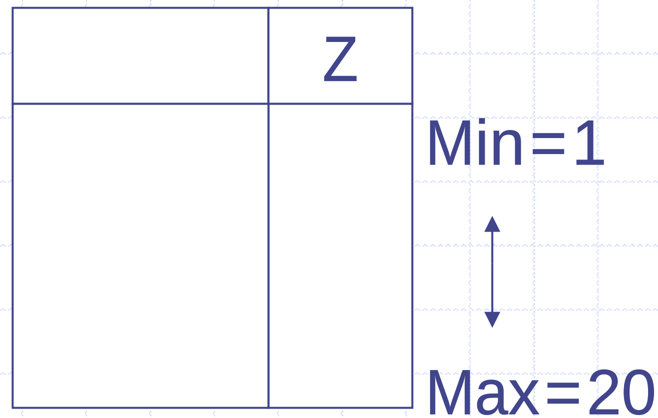
- 书上介绍的7.4.3 P369大概预测方法：

T(W) = T(R) /3

原因：直观上，这种条件选择趋向选择少的集合。

- Solution # 2: Estimate values in range

<u>Example</u> R



$V(R,Z)=20$

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

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

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

Equivalently:

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

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

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

$T(W) = ?$

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

σ操作预测结果元组大小的总结

- $W = \sigma_{z = val}(R)$
-
  - $T(W) = \dfrac{T(R)}{V(R,Z)}$
- $W = \sigma_{z \neq val}(R)$
  - $T(W) = T(R)$

  Or $T(W) = T(R)(V(R,z)-1)/V(R,z)$

- $W = \sigma_{z > val}(R)$ or $W = \sigma_{z < val}(R)$

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

  Or $T(W) = T(R)/3$

Example 1

- Given a relation R(a,b,c), S= $\sigma_{a = 10 \text{ and } b<20}$ (R).
- And T(R) = 10,000, V(R,a) = 50.
- Then T(S) =?

- T(S) =T(R) /(50*3)=67
- Or T(S) =T(R) /(50*2)=100
- Or T(S) =T(R) /50*f=200*f= 200*20/50=80

Example 2

- Given a relation R(a,b), S= $\sigma_{a = 10 \text{ or } b<20}$ (R).
- And T(R) = 10,000, V(R,a) = 50.
- Then T(S) =?

- M1= T(R)/V(R,a)=200
- M2 =T(R)/3 = 3333
- So T(S) =200 + 3333= 3533
- Another method:
- T(S) =T(R) *(1-(1-m1/n)(1-m2/n))
- =10000*(1-(1-200/10000)(1-3333/10000) = 3466

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(if there exists common attribute; if not, the result is null)

Case 2

$W = R1 \bowtie R2 \qquad X \cap Y = A$

R1

A	B	C

R2

A	D

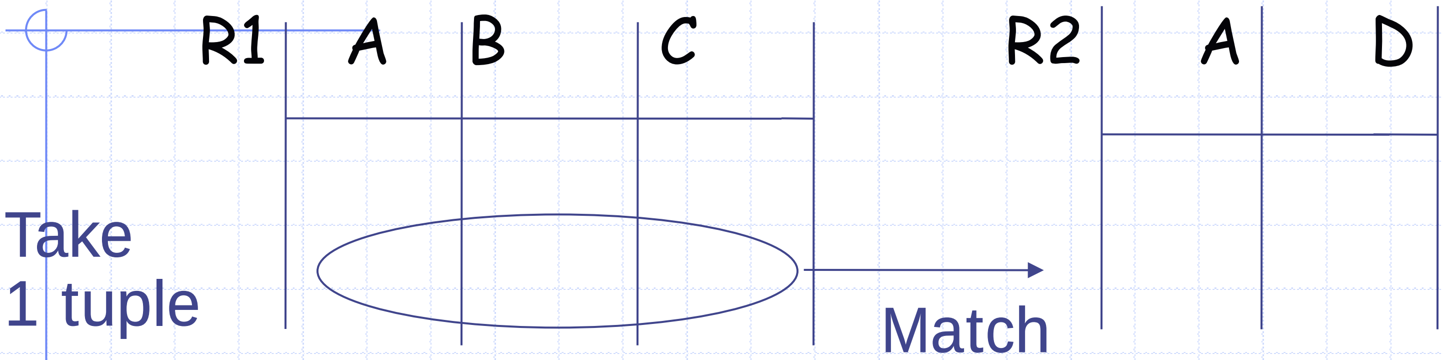
Assumption:

$V(R1,A) \leq V(R2,A) \Rightarrow$ 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

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



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

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

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

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

[A is common attribute]

In general $W = R1 \bowtie R2$

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

Give an Example

Example 3

- Given R(a,b), S(b,c), U(c,d)
- T(R)=1000; T(S)=2000;T(U)=5000
- V(R,b)=20; V(S,b)=50; V(S,c)=100;V(U,c) =500

- Then, $T(R \bowtie S \bowtie U) = ?$

1: $(R \bowtie S) \bowtie U$

2: $T(R \bowtie S) = T(R)T(S)/max(V(R,b), V(S,b))$

$= 1000*2000/50 = 40,000$

3: $T(R \bowtie S \bowtie U)$

$= T(R \bowtie S)T(U)/max(V(R \bowtie S, c), V(U,c))$

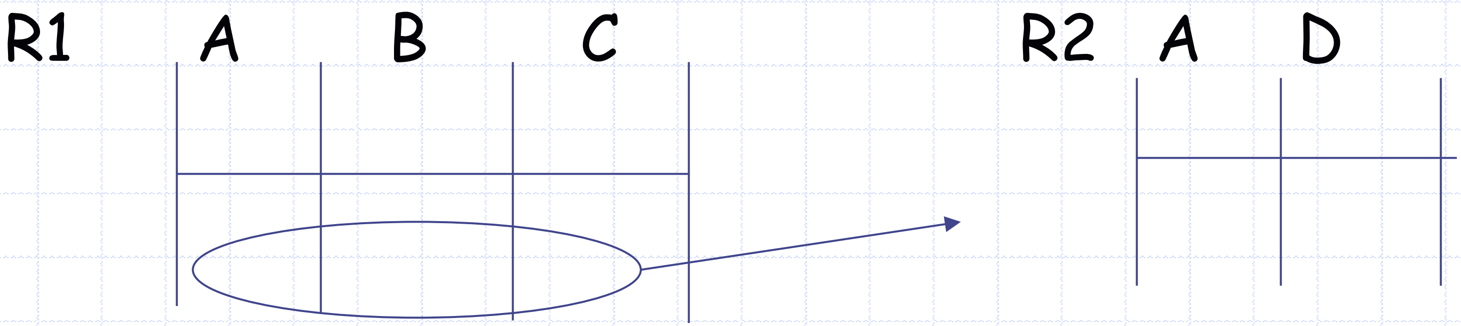
$= 40,000*5000/max(100,500) = 400,000$

You can also calculate by $R \bowtie (S \bowtie U)$

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

In all cases:

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

size of attribute *A*

Using similar ideas, we can estimate sizes of:

$\Pi_{AB}(R)$ ..... Sec. 7.4.2

$\sigma_{A=a \wedge B=b}(R)$ .... Sec. 7.4.3

$R \bowtie S$ with common attribs. A,B,C
Sec. 7.4.5涉及多个属性的自然连接

Union, intersection, diff, ... Sec. 7.4.7

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

E.g. $W = [\underbrace{\sigma_{A=a}(R1)}_{\text{Treat as relation } U}] \bowtie R2$

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

$V(U, A) =$

$V(U, B) =$

$V(U, C) =$

$V(U, D) =$

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

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

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

$V(U,A) = 1$ $V(U,B) = 1$ $V(U,C) = \dfrac{V(R1,C)}{V(R1,A)}$

$V(U,D)$ 的取值在 1 或者 $\dfrac{V(R1,D)}{V(R1,A)}$ 之间的某个值

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

$V(U,A) = 1$

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

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 7.4.4]

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

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

V(U,A) =?

V(U,B) =?

V(U,C) =?

Partial Result: $U = R \bowtie S$

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

$V(U,A) = 50$

$V(U,B) = 100$

$V(U,C) = 300$

$Z = U \bowtie R3$

$T(Z)=?$

$V(Z,A)=?$

$V(Z,B)=?$

...

$Z = U \bowtie R3$

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

$V(Z,A) = 50$

$V(Z,B) = 100$

$V(Z,C) = 90$

$V(Z,D) = 500$

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…
- Generate and compare plans

Exercise

- P351 7.2.1
- P353 7.2.6
- P379 7.4.1 a,b,g

- P382 7.5.1 Histogram for estimation