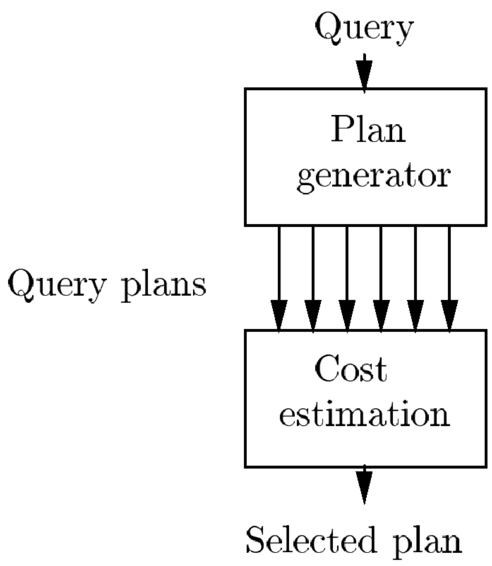
Query Processing

Overview Query Processing



Query Plans

Choose operations, e.g., σ, $\bowtie$

Order operations.

Detailed strategy of operations, e.g.:

- Join method.
- Pipelining: consume result of one operation by another, to avoid temporary storage on disk.
- Use of indexes?
- Sort intermediate results?

We focus on relational systems

Implementation of DBMS
WS 23/24
Frankfurt UAS
Prof. Dr. Justus Klingemann

Example

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

Implementation of DBMS
WS 23/24
Frankfurt UAS
Prof. Dr. Justus Klingemann

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 the 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 to Describe Plans

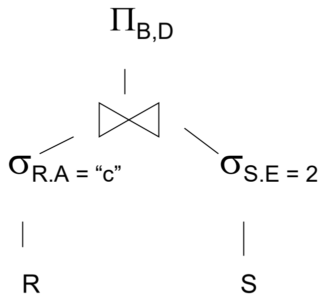
Ex: Plan I

$$\Pi_{B,D}$$
$$\sigma_{R.A=\text{“c”} \wedge S.E=2 \wedge R.C=S.C}$$
$$\times$$
$$R \qquad S$$

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

Another Plan

Plan II



$\bowtie$ natural join

R

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

σ (R)

A	B	C
c	2	10

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 with S.E $\neq$ 2

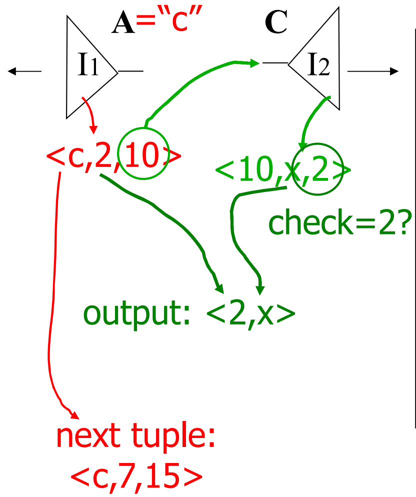
(4) Join matching R,S tuples,

(5) Project B,D attributes and place in result

Using the index on R.A to retrieve tuples where R.A = "c".
Using the index on S.C to find matching tuples in S for each R.C value found.
Eliminating tuples in S where S.E not 2.
Joining matching tuples from R and S.
Projecting attributes B and D into the result.

R

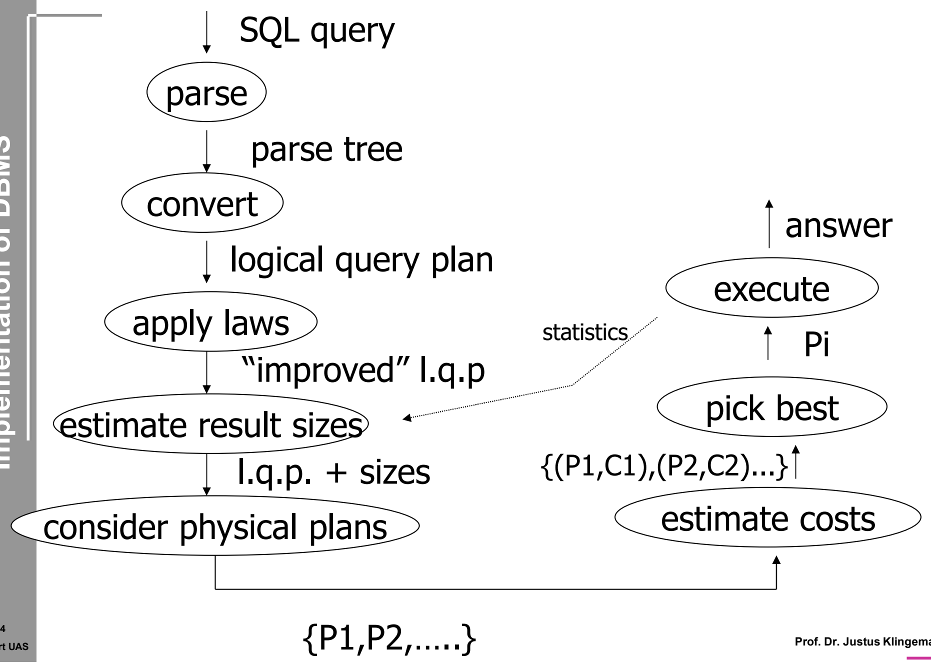
A	B	C
a	1	10
b	1	20
c	2	10
d	2	35
e	3	45



S

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

Overview of Query Optimization



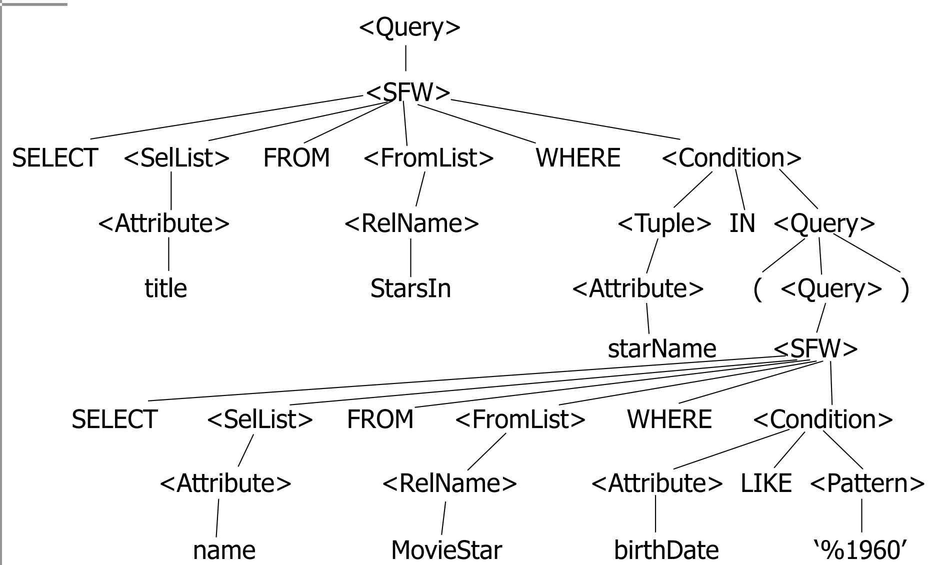
Example: SQL Query

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

(Find the movies with stars born in 1960)

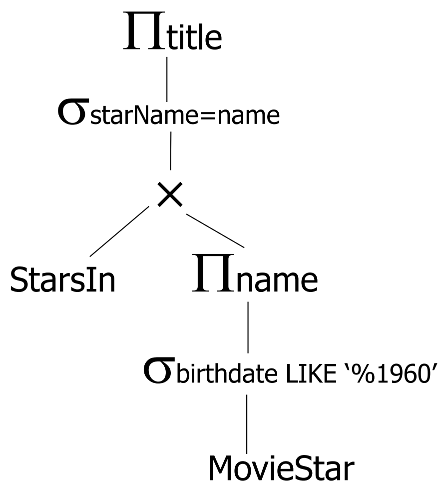
Implementation of DBMS
WS 23/24
Frankfurt UAS
Prof. Dr. Justus Klingemann

Example: Parse Tree



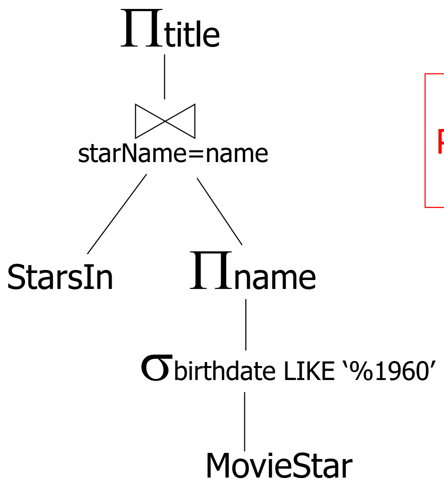
Implementation of DBMS
WS 23/24
Frankfurt UAS
Prof. Dr. Justus Klingemann

Example: Logical Query Plan



Implementation of DBMS
WS 23/24
Frankfurt UAS
Prof. Dr. Justus Klingemann

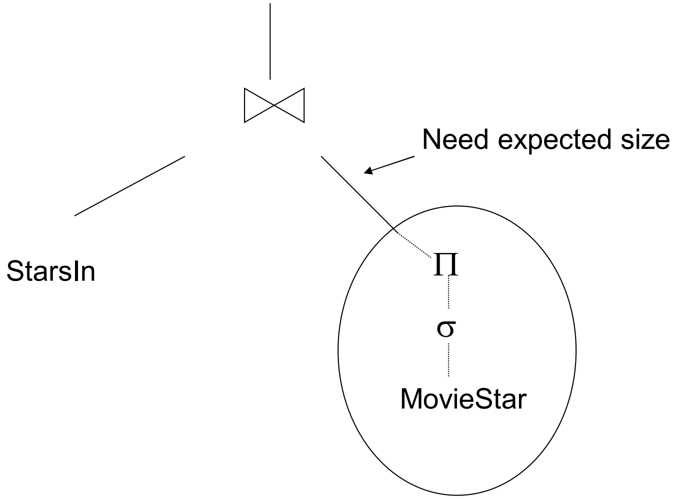
Example: Improved Locical Query Plan



Question:
Push project to
StarsIn?

Implementation of DBMS

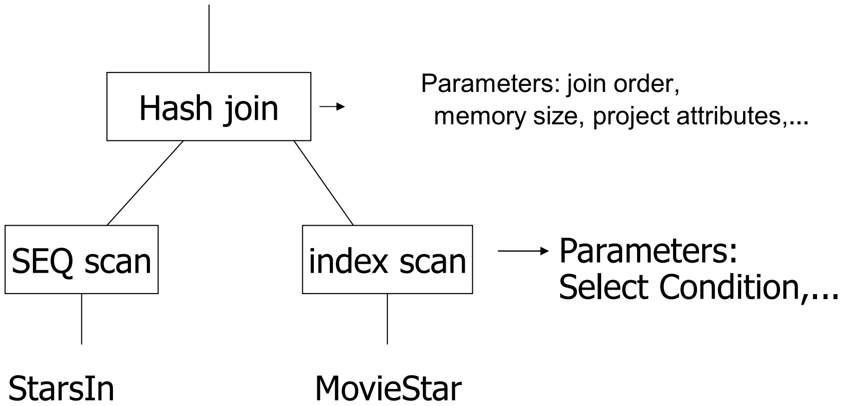
Example: Estimate Result Sizes



WS 23/24
Frankfurt UAS
Prof. Dr. Justus Klingemann

Implementation of DBMS

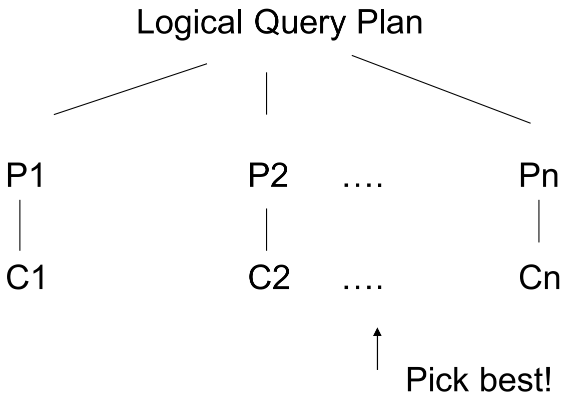
Example: One Physical Plan



WS 23/24
Frankfurt UAS
Prof. Dr. Justus Klingemann

Implementation of DBMS

Example: Estimate Costs



WS 23/24
Frankfurt UAS
Prof. Dr. Justus Klingemann

Algebraic Transformations

Generating Plans

- Start with query definition.
  - A plan, but usually a terrible one.
- Apply algebraic transformations to find other plans.
- Relational algebra is a good start, but we need also to consider: GROUP BY, duplicate elimination, HAVING, ORDER BY.

Algebraic Transformations

- Rules give equivalent expressions. meaning that whatever relations are substituted for variables, the results are the same.

Implementation of DBMS

Rules: Natural joins & cross products & union

$R \bowtie S \quad = \quad S \bowtie R$

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

$R \times S = S \times R$

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

$R \cup S = S \cup R$

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

But beware of thetajoin (join condition different from =)

- associative law does not hold.

Implementation of DBMS

Rules: Selects

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

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

Implementation of DBMS

Rules: Project

Let: X = set of attributes

Y = set of attributes

XY = X U Y

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

Implementation of DBMS

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: $\sigma + \bowtie$ combined

Some rules can be derived:

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

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

$$\sigma_{p \vee q} (R \bowtie S) = \left[(\sigma_p R) \bowtie S\right] \cup \left[R \bowtie (\sigma_q S)\right]$$

Rules: π, σ combined

Let x = subset of R attributes

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

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

Rules: π, $\bowtie$ combined

Let x = subset of R attributes

y = subset of S attributes

z = intersection of R, S attributes

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

Rules: π, $\bowtie$, and σ combined

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

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

$z' = z \cup \{\text{attributes used in } P\}$

Rules: σ, U combined

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

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

Which are "good" transformations?

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

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

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

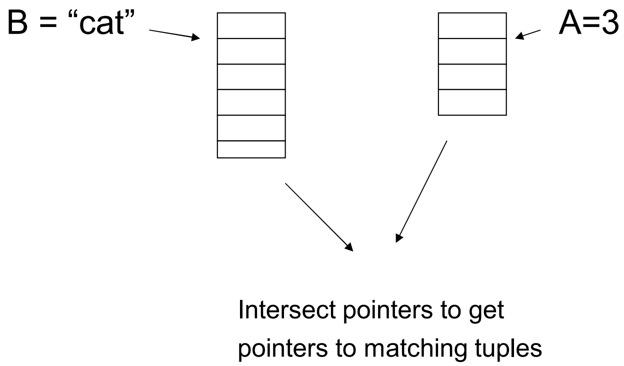
$\pi_x[\sigma_p(R)] \rightarrow \pi_x\{\sigma_p[\pi_{xz}(R)]\}$

Conventional wisdom: do projects early

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

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

What if we have A, B indexes?



Bottom line

There is no transformation that is good in any case

Usually good: early selections

- Variant: move selection up to the root and then down on multiple path

Selections Should Go Up Then Down

StarsIn(title, year, starName)

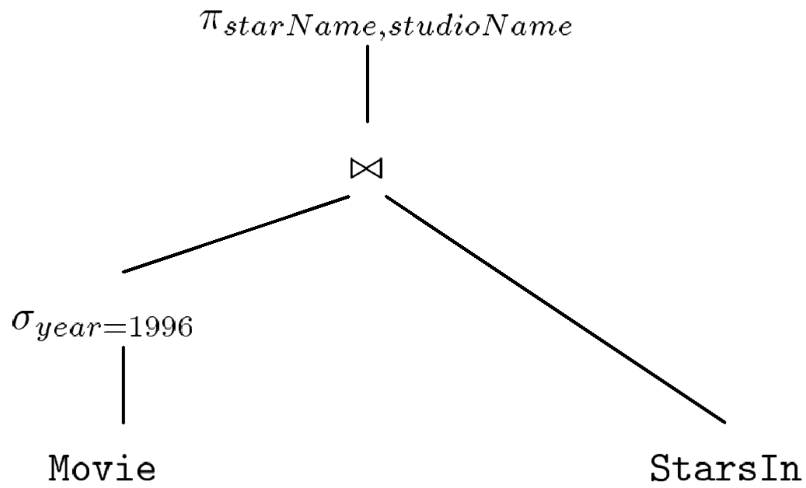
Movie(title, year, studioName)

```
CREATE VIEW MoviesOf1996 AS
SELECT *
FROM Movie
WHERE year = 1996;
```

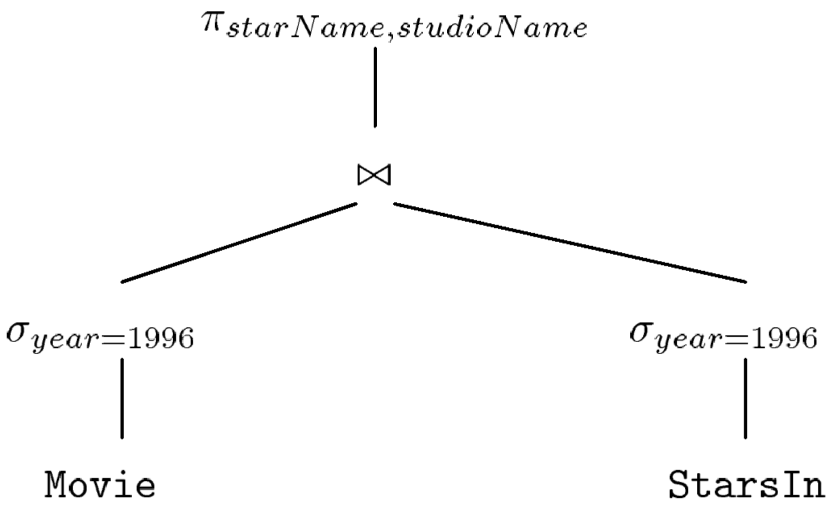
```
SELECT starName, studioName
FROM MoviesOf1996 NATURAL JOIN StarsIn;
```


Example Cont.

Initial query:



Probably Better



Estimating the Cost of a Query Plan

Goal is to count disk I/O's.

But we first have to estimate sizes of intermediate results.

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
- DOM (R, A): # possible distinct values for attribute A (size of domain for 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 S(R) = 37

V(R,A) = 3 V(R,C) = 5

V(R,B) = 1 V(R,D) = 4

Size estimates for W = R1 x R2

T(W) = T(R1) $\times$ T(R2)

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

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

S(W) = S(R)

T(W) = ?

Estimate Depends on Assumption

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

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

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

Alternate Assumption

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

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

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

Selections Involving Inequality

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

T(W) = ?

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

- Assumption: All split values are equally likely

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

- Assumption: Queries involving inequality ask more likely for a small fraction of possible tuples
- This assumption is usually preferred

Selections Involving Inequality (cont.)

Solution # 3: Estimate values in range

<u>Example</u> R

	Z

Min=1

Max=20

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

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

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