CS525: Advanced Database Organization

Notes 6: Query Processing Overview

Yousef M. Elmehdwi

Department of Computer Science

Illinois Institute of Technology

yelmehdwi@iit.edu

October 4, 2018

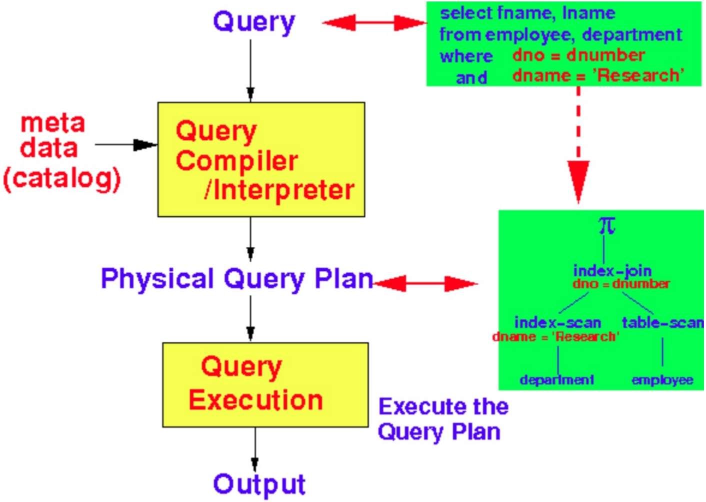
Slides: adapted from a courses taught by Hector Garcia-Molina, Stanford, & Shun Yan Cheung, Emory University

Where we are

- How a DBMS processes queries and the methods it uses to optimize their performance.

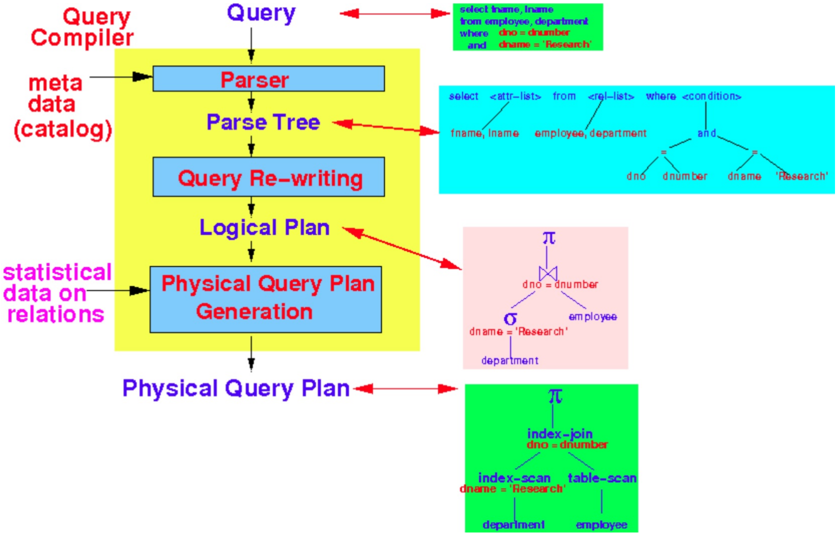
Query Processing: The major parts of the query processor

- Steps needed to process a query (SQL command)



Query Processing

- The **Query Compiler** consists of 3 major steps



Query Processing: Example

```
SELECT B,D
FROM R,S
WHERE R.A='c' and S.E=2 and R.c=S.c
```

Query Processing: Example

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

Query Processing: Example

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

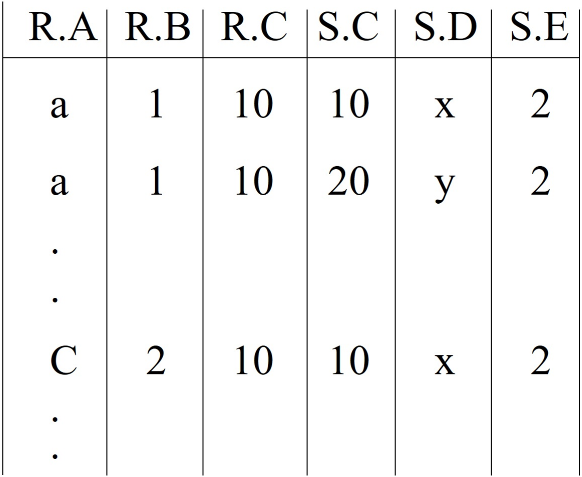
B	D
2	x

How do we execute query?

- One idea
  - Scan relations
  - Do Cartesian product
  - Select tuples
  - Do projection

Query Processing: Example

$R \times S$



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
.					
.					

Query Processing: Example

RXS

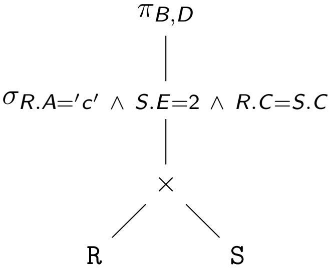
R.A	R.B	R.C	S.C	S.D	S.E
a	1	10	10	x	2
a	1	10	20	y	2
.					
.					
C	2	10	10	x	2
.					
.					

Bingo! →

Got one...

Relational Algebra

- can be used to describe plans
- Example: Plan I: Initial query plan constructed directly from the query.



$\pi_{B,D}$

$\sigma_{R.A='c' \land S.E=2 \land R.C=S.C}$

$\times$

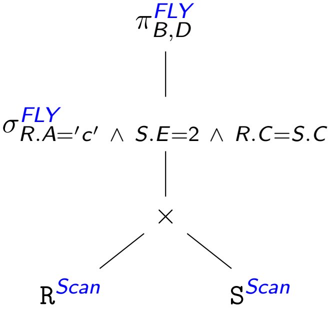
R

S

- FROM expressed by a `product`, WHERE by a `selection` above it, Select by a `projection`

Relational Algebra

- can be used to describe plans
- Example: Plan I: Initial query plan constructed directly from the query.



$\pi^{FLY}_{B,D}$

$\sigma^{FLY}_{R.A='c' \,\wedge\, S.E=2 \,\wedge\, R.C=S.C}$

$\times$

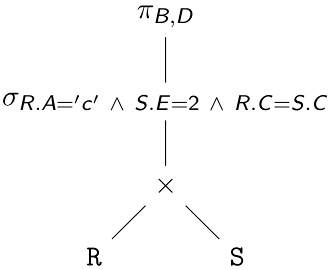
$\mathtt{R}^{Scan}$ $\mathtt{S}^{Scan}$

1. Scan `R`
2. For each tuple `r` of `R` scan `S`
3. For each tuple `r,s`, where `s` in `S`, select and project on the fly

- OR: $\pi^{FLY}_{B,D}\left[\sigma^{FLY}_{R.A='c' \,\wedge\, S.E=2 \,\wedge\, R.C=S.C}(R^{Scan} \times S^{Scan})\right]$

“FLY” and “SCAN” are the defaults

- Example: Plan I: Initial query plan constructed directly from the query.



$\pi_{B,D}$

$\sigma_{R.A='c' \ \wedge \ S.E=2 \ \wedge \ R.C=S.C}$

$\times$

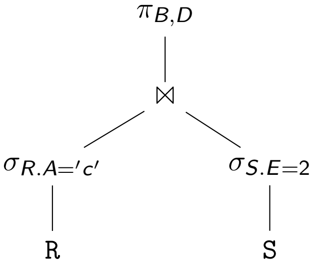
R

S

- OR: $\pi_{B,D}\left[\sigma_{R.A='c' \ \wedge \ S.E=2 \ \wedge \ R.C=S.C}(R \times S)\right]$

Another idea

- Example: Plan II: Scan `R` and `S`, perform on the fly selections, do hash join, project



$\pi_{B,D}$

$\bowtie$

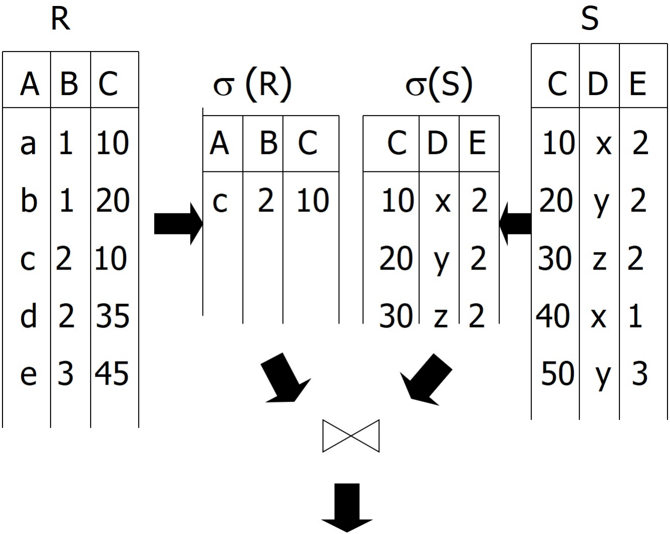
$\sigma_{R.A='c'}$

$\sigma_{S.E=2}$

R

S

Another idea



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

$\bowtie$

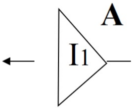
Plan III: Use R.A and S.C Indexes

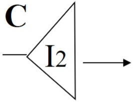
1. Use `R.A index` to select `R` tuples with `R.A = 'c'`
2. For each `R.C` value found, use `S.C index` to find matching tuples
3. Eliminate `S` tuples `S.E` $\neq$ `2`
4. Join matching `R,S` tuples, project `B,D` attributes and place in result

Plan III: Use R.A and S.C Indexes

R

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

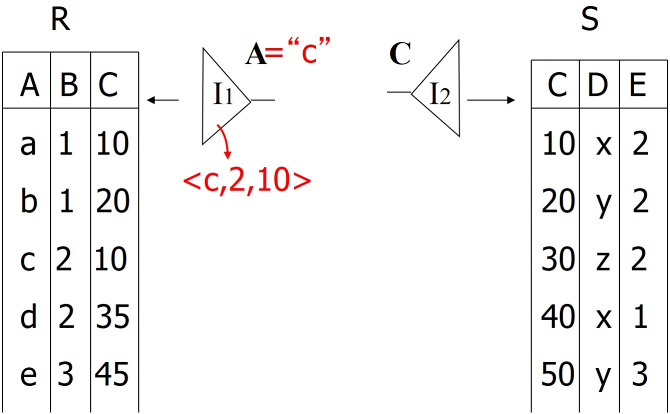




S

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

Plan III: Use R.A and S.C Indexes



R

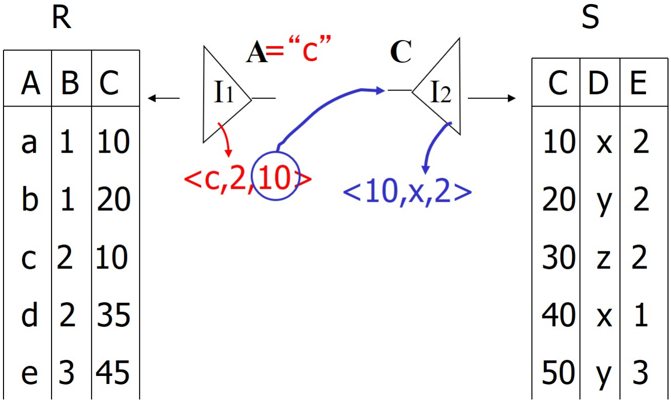
A	B	C
a	1	10
b	1	20
c	2	10
d	2	35
e	3	45

S

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

1. Use `R.A index` to select `R` tuples with `R.A = 'c'`

Plan III: Use R.A and S.C Indexes



R

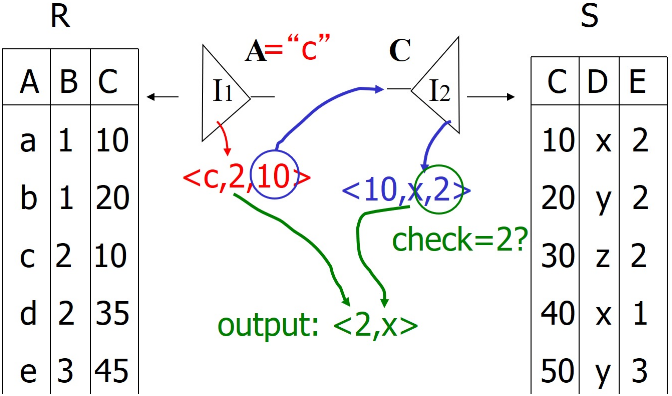
A	B	C
a	1	10
b	1	20
c	2	10
d	2	35
e	3	45

S

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

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

Plan III: Use R.A and S.C Indexes



R

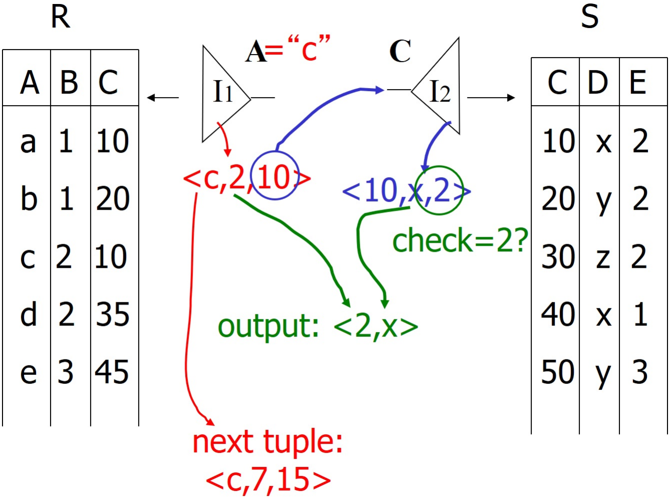
A	B	C
a	1	10
b	1	20
c	2	10
d	2	35
e	3	45

S

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

3. Eliminate S tuples S.E $\neq$ 2
4. Join matching R,S tuples, project B,D attributes and place in result

Plan III: Use R.A and S.C Indexes



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

<c,2,10>

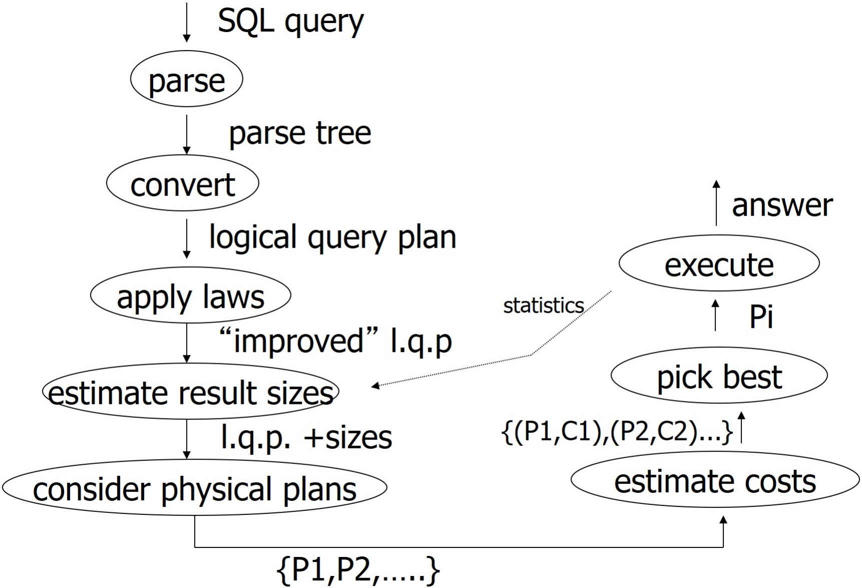
<10,x,2>

check=2?

output: <2,x>

next tuple:
<c,7,15>

Overview of Query Optimization

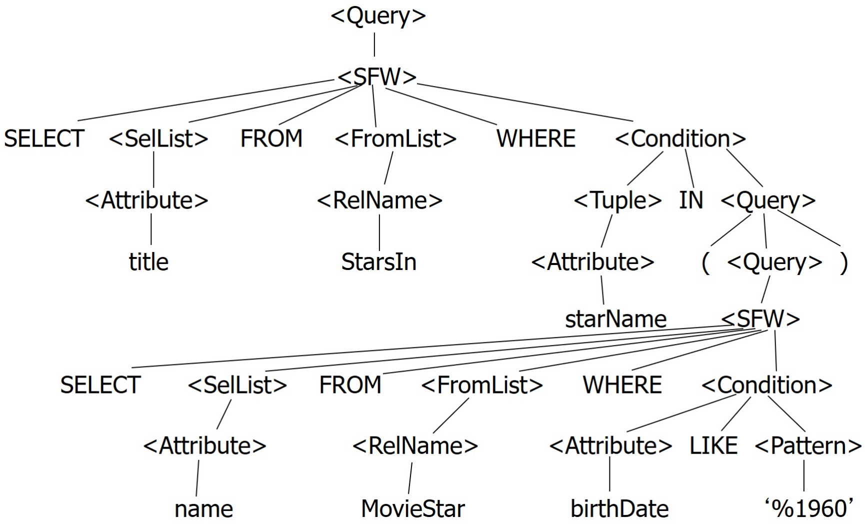


Example: SQL query

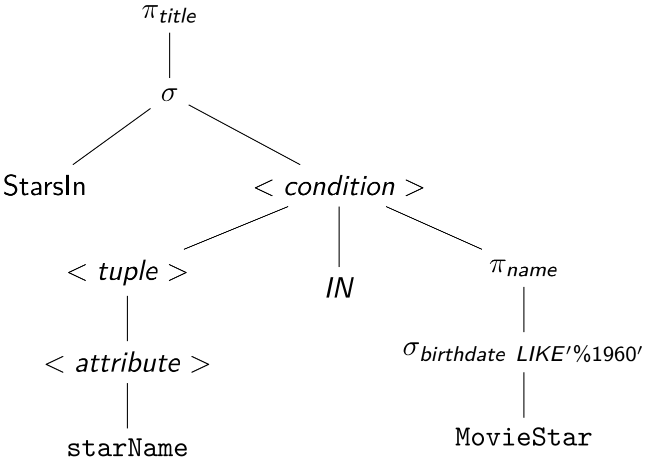
```
SELECT title
FROM   StarsIn
WHERE starName IN (SELECT name
                   FROM MovieStar
                   WHERE birthdate LIKE ‘%1960‘ );
```

- Find the movies with stars born in 1960

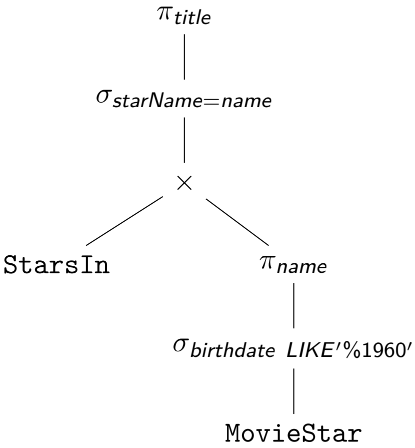
Example: Parse Tree



Example: Generating Relational Algebra



Example: Logical Query Plan



π_{title}

$\sigma_{starName=name}$

$\times$

StarsIn

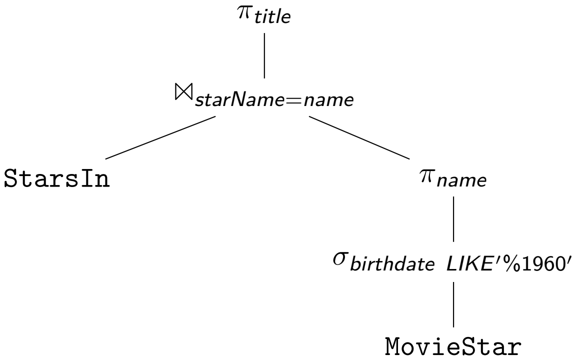
π_{name}

$\sigma_{birthdate\ LIKE'\%1960'}$

MovieStar

- May consider "IN" elimination as a rewriting in the logical plan generator or may consider it a task of the converter

Example: Improved Logical Query Plan



π_{title}

$\bowtie_{starName=name}$

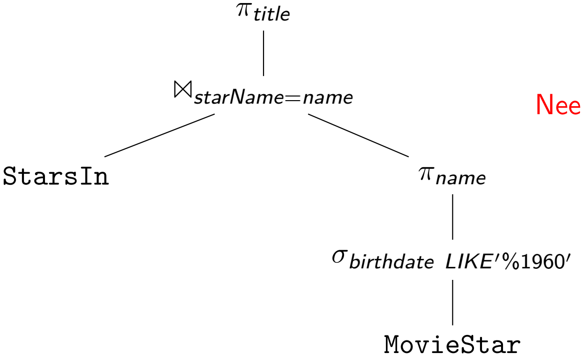
StarsIn

π_{name}

$\sigma_{birthdate\ LIKE'\%1960'}$

MovieStar

Result sizes are important for selecting physical plans



π_{title}

$\bowtie_{starName=name}$

StarsIn

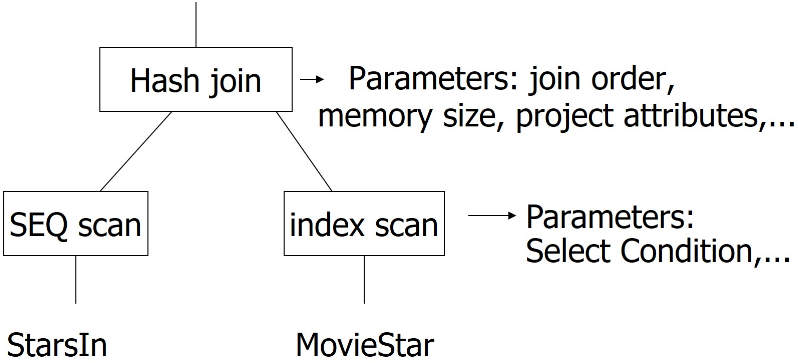
π_{name}

$\sigma_{birthdate\ LIKE '\%1960'}$

MovieStar

Need expected size

Example: One Physical Plan



Hash join → Parameters: join order, memory size, project attributes,...

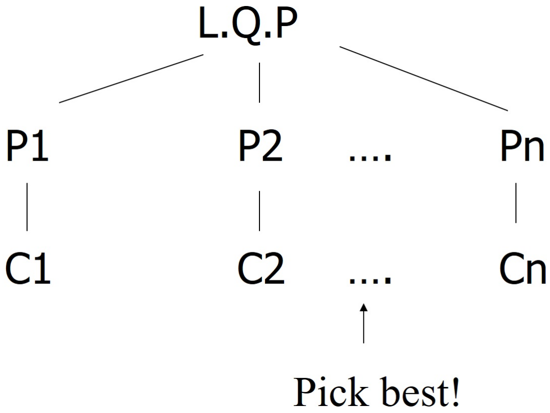
SEQ scan

index scan → Parameters: Select Condition,...

StarsIn

MovieStar

Example: One Physical Plan


L.Q.P
P1
P2
....
Pn
C1
C2
....
Cn
Pick best!