COL 362 & COL 632

Processing SQL – plan generation

03 Feb 2023

Assignment 1

- **Remember that we will stop answering questions on Piazza 10 hours before the deadline – i.e., at 2PM today.**
- Please remember that TAs are also humans (some of you will be TAs in the future), and have other courses
  - Be respectful, don’t expect them to respond to you within seconds of posting.
- Check the clarification thread if your questions are already answered before you post
- Follow only the clarifications given on Piazza

Example (1/5)

Actors

Name	Age	Addr
Priyanka Chopra	36	Mumbai
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Movies

Name	Year	Title
Priyanka Chopra	2011	Don-II
Anthony Hopkins	2011	MI-IV
Bill Nighy	2009	Valkyrie
Abhishek Bachchan	2010	Raavan

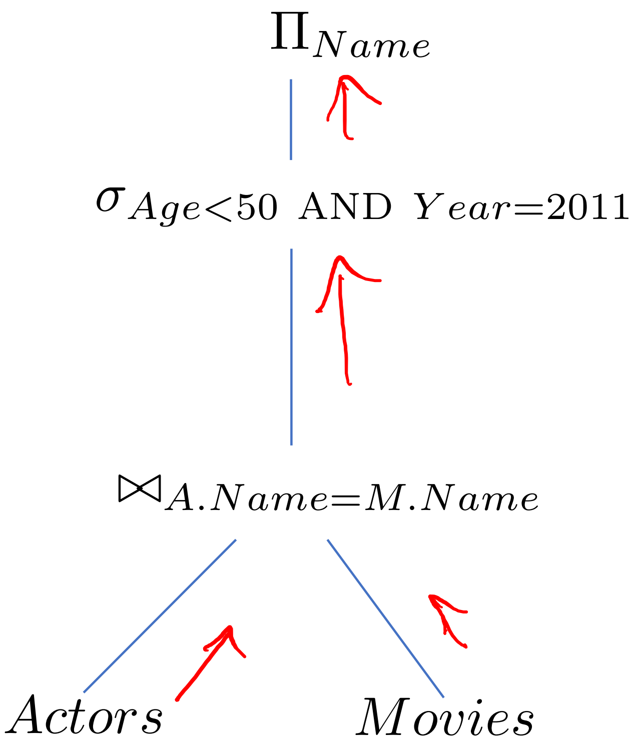
Return the names of actors below the age of 50 who have acted in a movie in 2011

$$\Pi_{Name}(\underline{\sigma_{Age<50 \text{ AND } Year=2011}}(\underline{Actors \bowtie_{A.Name=M.Name} Movies}))$$

$$\begin{aligned} Allmovies &= Actors \bowtie_{A.Name=M.Name} Movies \\ Movies1 &= \sigma_{Age<50 \text{ AND } Year=2011}(AllMovies) \\ Result &= \Pi_{Name}(Movies1) \end{aligned}$$

Example (2/5)

$$\Pi_{Name}(\sigma_{Age<50 \text{ AND } Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies))$$



Π_{Name}

$\sigma_{Age<50 \text{ AND } Year=2011}$

$\bowtie_{A.Name=M.Name}$

$Actors$ $Movies$

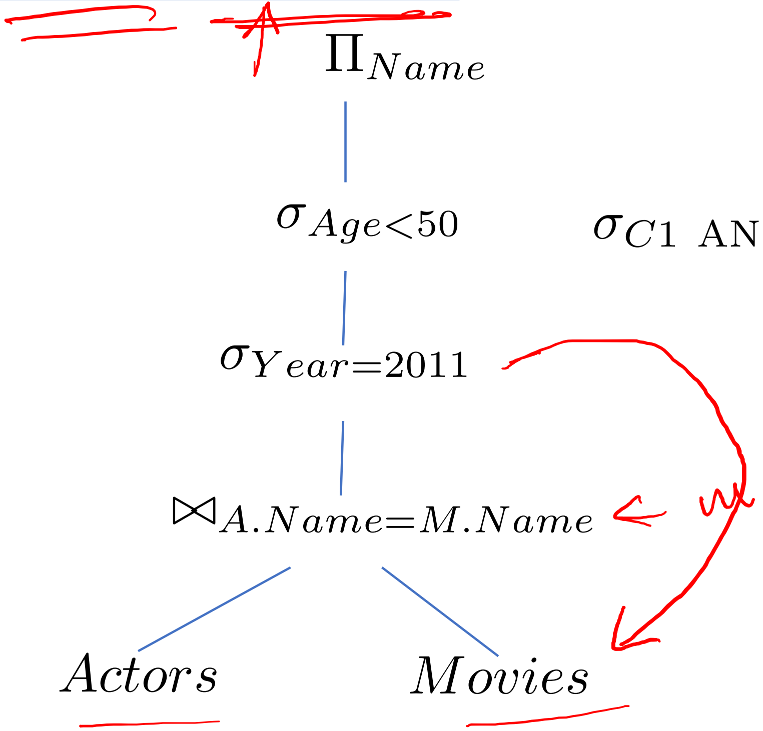
Example (3/5)

2m

m+1

$$\Pi_{Name}(\sigma_{Age<50 \text{ AND } Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies))$$

$$\Pi_{Name}(\sigma_{Age<50}(\sigma_{Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies)))$$



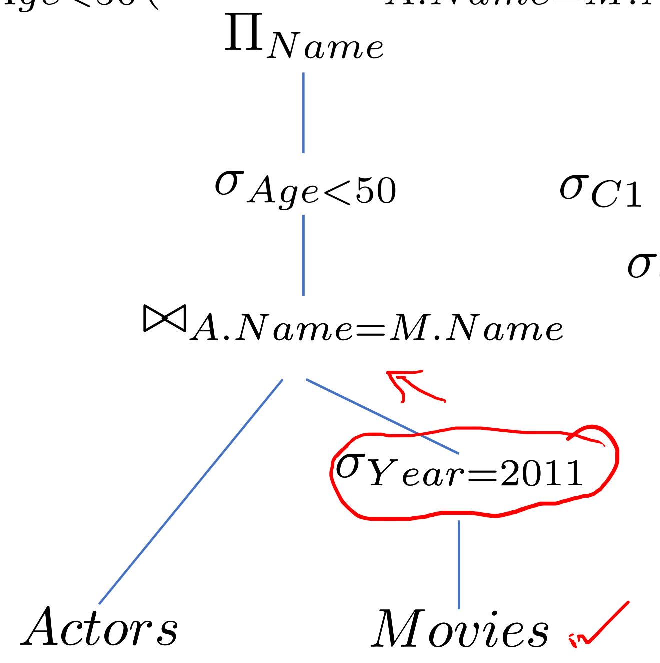
$$\sigma_{C1 \text{ AND } C2}(S) = \sigma_{C1}(\sigma_{C2}(S))$$

Example (4/5)

$$\Pi_{Name}(\sigma_{Age<50 \text{ AND } Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies))$$

$$\Pi_{Name}(\sigma_{Age<50}(\sigma_{Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies)))$$

$$\Pi_{Name}(\sigma_{Age<50}(Actors \bowtie_{A.Name=M.Name} (\underline{\sigma_{Year=2011}}(Movies))))$$



$$\sigma_{C1 \text{ AND } C2}(S) = \sigma_{C1}(\sigma_{C2}(S))$$

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

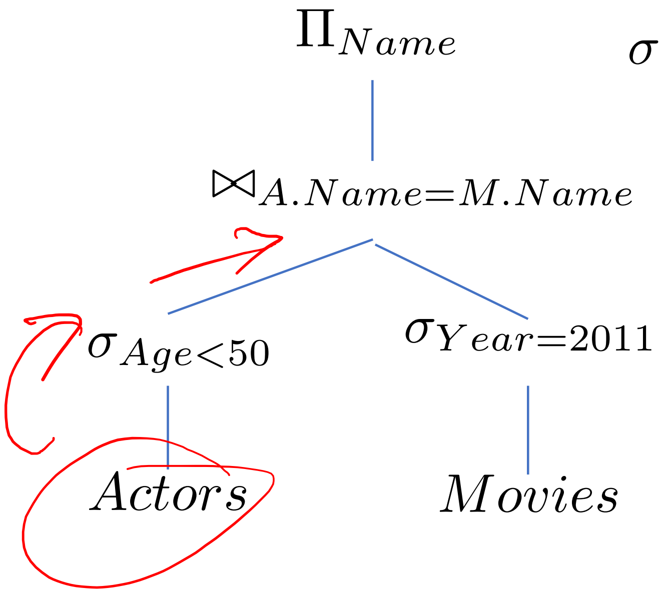
Example (5/5)

$$\Pi_{Name}(\sigma_{Age<50 \text{ AND } Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies))$$

$$\Pi_{Name}(\sigma_{Age<50}(\sigma_{Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies)))$$

$$\Pi_{Name}(\sigma_{Age<50}(Actors \bowtie_{A.Name=M.Name} (\sigma_{Year=2011}(Movies))))$$

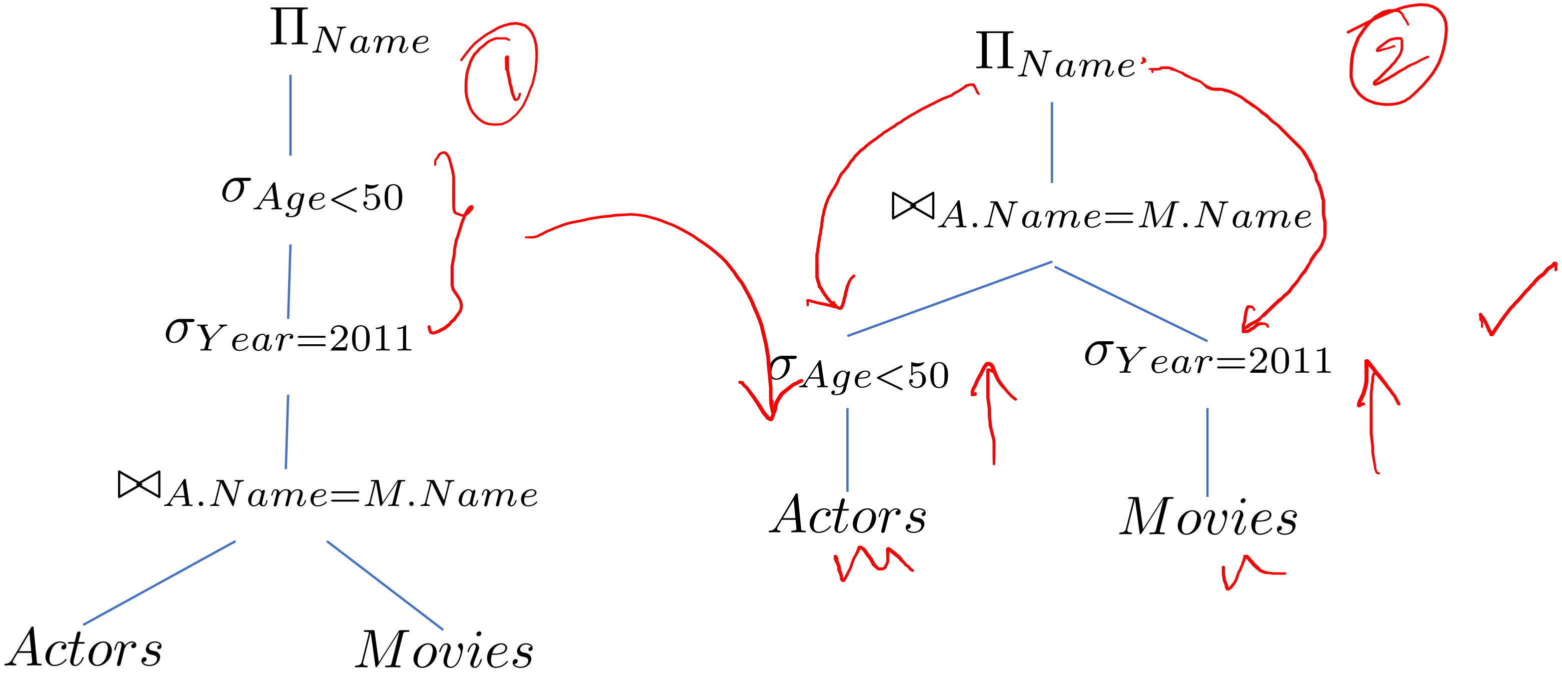
$$\Pi_{Name}(\sigma_{Age<50}(Actors) \bowtie_{A.Name=M.Name} (\sigma_{Year=2011}(Movies)))$$



$$\sigma_{C1 \text{ AND } C2}(S) = \sigma_{C1}(\sigma_{C2}(S))$$

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

Which is more efficient?



1

Π_{Name}

$\sigma_{Age<50}$

$\sigma_{Year=2011}$

$\bowtie_{A.Name=M.Name}$

$Actors$ $Movies$

2

Π_{Name}

$\bowtie_{A.Name=M.Name}$

$\sigma_{Age<50}$ $\sigma_{Year=2011}$

$Actors$ $Movies$

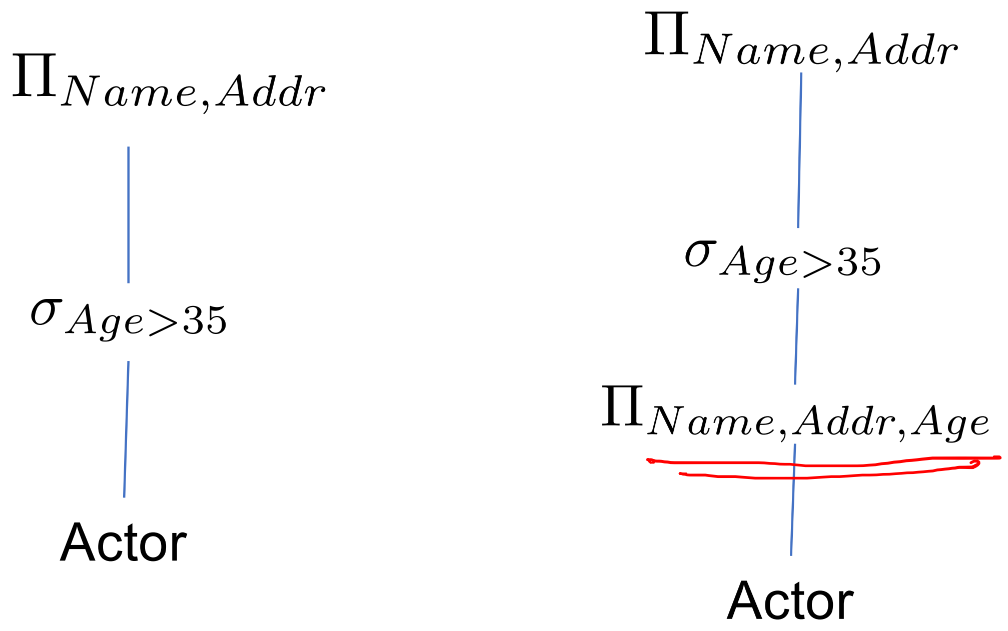
✓

Pushing selections lower in the tree, almost always results in a more efficient plan

More Algebraic Laws (1/2)

$$\pi_L(T \bowtie S) = \pi_L(\pi_M(T) \bowtie \pi_N(S))$$

$$\pi_L(\sigma_C(S)) = \pi_L(\sigma_C(\pi_M(S)))$$



$$\Pi_{Name,Addr}$$

$$\sigma_{Age>35}$$

Actor

$$\Pi_{Name,Addr}$$

$$\sigma_{Age>35}$$

$$\Pi_{Name,Addr,Age}$$

Actor

More Algebraic Laws (2/2)

$$\delta(S) = S \quad \longleftarrow \quad \text{When?}$$

$$\delta(S \bowtie T) = \boxed{\delta(S)} \bowtie \delta(T)$$

$$\delta(\sigma_C(S)) = \sigma_C(\delta(S))$$

$$\delta(\underset{\uparrow}{\gamma_L}(S)) = \gamma_L(S)$$

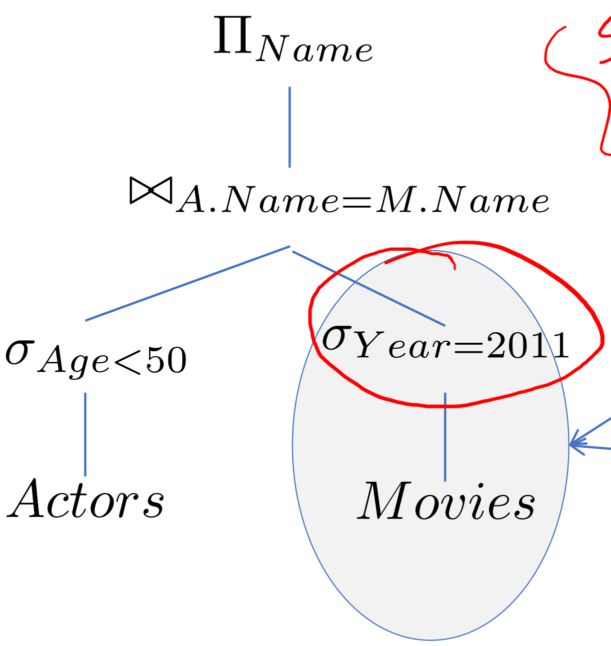
General Rules

- Minimize the number and size of tuples you need to process
  - Push selections low
  - Push projections low
  - Push duplicate eliminations low
- Ordering of operations (we will study this)
  - Which of k selections first?
  - Which of k joins first?

Physical Plans

Physical operators – Selection

spilling of results to disk

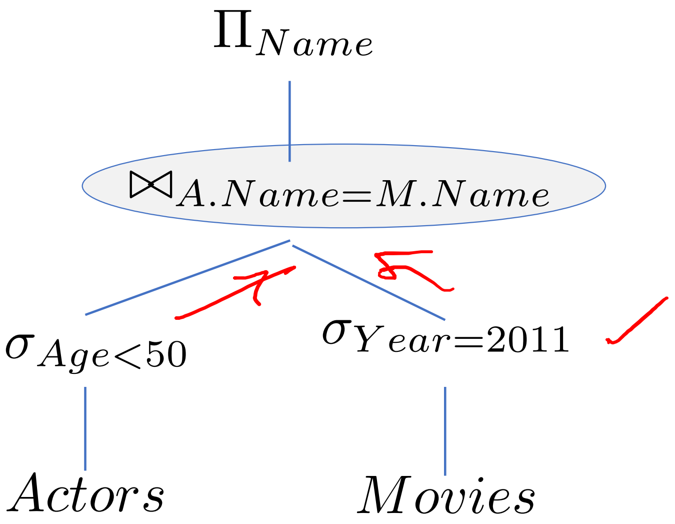


- SCAN the tuples in Movies one by one
- Retain tuples which satisfy the condition

- If an index is present on Year, then perform an INDEX-SCAN
- Fetch tuples which satisfy the condition

The output of the operator: disk or memory?

Physical Operators – Joins (1/2)



Nested-loop Join (R,S)

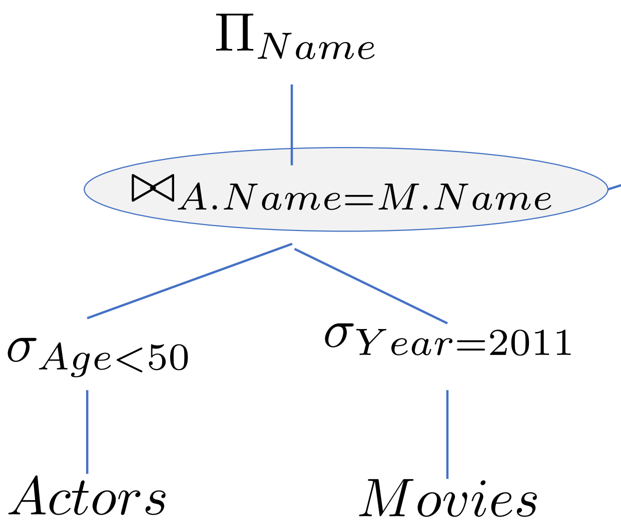
```
iterate over $r_i$ in R
  iterate over $s_i$ in S
     if (join-condition satisfied)
        output  join($r_i$,$s_i$)
  end iteration
end iteration
```

Hash Join (R,S)

```
iterate over $r_i$ in R
  hash $r_i$
iterate over $s_i$ in S
  hash $s_i$
join tuples in same bucket
```

When won’t this work?

Physical Operators – Joins (2/2)



Sort-Merge Join (R,S)
Sort R
Sort S
Merge based on join condition

Index-based Join (R,S)
Indexes provide sorted access

What kind of indexes are ideal?