L22: The Relational Model (continued)

CS3200 Database design (sp18 s2)

https://course.ccs.neu.edu/cs3200sp18s2/

4/5/2018

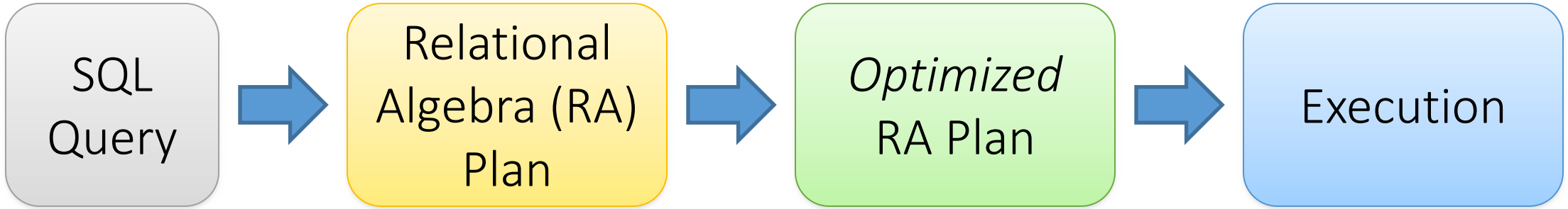
Announcements!

- Please pick up your exam if you have not yet
- HW6 will include Redis and MongoDB exercises with minimal install overhead (still in preparation)
- Final class calendar
- Outline today
  - Relational algebra
  - Optimization
  - NoSQL (start, continuing next time)

		Query Processing and Database Internals		
17	M Mar 19	**Exam 2** I/O Cost Models & Merge Sort	GUW Ch 11.4	
18	R Mar 22	I/O Cost Models & External Sort	GUW Ch 11.4	Q8
19	M Mar 26	Indexing and B+ trees	GUW Ch 13.1-13.3	
20	R Mar 29	Joins 1	GUW Ch 15.9	
21	M Apr 2	Joins 2, Relational Algebra	GUW Ch 2 and 16.3	HW5
22	R Apr 5	Relational Algebra, Query Optimization, NoSQL	GUW Ch 5	P2 (R 4/5), Q9 (FR 4/6)
23	M Apr 9	NoSQL	GUW Ch 8 and 14	
		NoSQL		
24	R Apr 12	NoSQL, Class Review and Course Evaluation		Q10 (optional)
	M Apr 16	No class: Patriot's day		HW6 (TU 4/17)
	R Apr 19	No class: Reading day		Optional PPTX (Wed 4/18)
	M Apr 23	**Exam 3** (1-3pm, location TBD)		

RDBMS Architecture

- How does a SQL engine work ?



Declarative query (from user)

Translate to relational algebra expresson

Find logically equivalent- but more efficient- RA expression

Execute each operator of the optimized plan!

RDBMS Architecture

- How does a SQL engine work ?

Relational Algebra allows us to translate declarative (SQL) queries into precise and optimizable expressions!

Relational Algebra (RA)

- Five basic operators:
  1. Selection: σ
  2. Projection: Π
  3. Cartesian Product: $\times$
  4. Union: $\cup$
  5. Difference: -
- Derived or auxiliary operators:
  - Intersection, complement
  - Joins (natural, equi-join, theta join, semi-join)
  - Renaming: ρ
  - Division

We'll look at these first!

And also at one example of a derived operator (natural join) and a special operator (renaming)

Keep in mind: RA operates on sets!

- RDBMSs use **multisets**, however in relational algebra formalism we will consider sets!

- Also: we will consider the **named perspective**, where every attribute must have a unique name
  - →attribute order does not matter…

Now on to the basic RA operators…

1. Selection (σ)

- Returns all tuples which satisfy a condition
- Notation: $\sigma_c(R)$
- Examples
  - $\sigma_{Salary > 40000}$ (Employee)
  - $\sigma_{name = \text{“Smith”}}$ (Employee)
- The condition c can be =, <, ≤, >, ≥, <>

```
Students(sid,sname,gpa)
```

SQL:

```
SELECT *
FROM Students
WHERE gpa > 3.5;
```

RA:

$$\sigma_{gpa>3.5}(Students)$$

Another example:

SSN	Name	Salary
1234545	John	200000
5423341	Smith	600000
4352342	Fred	500000

$\sigma_{Salary > 40000}$ (Employee)

SSN	Name	Salary
5423341	Smith	600000
4352342	Fred	500000

2. Projection (Π)

- Eliminates columns, then removes duplicates
- Notation: $\Pi_{A1,\ldots,An}(R)$
- Example: project social-security number and names:
  - $\Pi_{SSN,\ Name}$ (Employee)
  - Output schema: Answer(SSN, Name)

```
Students(sid,sname,gpa)
```

SQL:

```
SELECT DISTINCT
   sname,
   gpa
FROM Students;
```

RA:

$$\Pi_{sname,gpa}(Students)$$

Another example:

SSN	Name	Salary
1234545	John	200000
5423341	John	600000
4352342	John	200000

Π_{SSN} (Employee)

SSN
1234545
5423341
4352342

Another example:

SSN	Name	Salary
1234545	John	200000
5423341	John	600000
4352342	John	200000

$\Pi_{\text{Name,Salary}}$ (Employee)

Name	Salary
John	200000
John	600000

Note that RA Operators are Compositional!

```
Students(sid,sname,gpa)
```

```
SELECT DISTINCT
   sname,
   gpa
FROM Students
WHERE gpa > 3.5;
```

How do we represent this query in RA?

$$\Pi_{sname,gpa}(\sigma_{gpa>3.5}(Students))$$

$$\sigma_{gpa>3.5}(\Pi_{sname,gpa}(Students))$$

Are these logically equivalent?

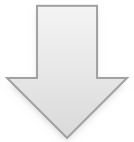
3. Cross-Product (×)

- Each tuple in R1 with each tuple in R2
- Notation: R1 × R2
- Example:
  - Employee × Dependents
- Rare in practice; mainly used to express joins

```
Students(sid,sname,gpa)
People(ssn,pname,address)
```

SQL:

```
SELECT *
FROM Students, People;
```



RA:

$$Students \times People$$

Another example:

People

ssn	pname	address
1234545	John	216 Rosse
5423341	Bob	217 Rosse

×

Students

sid	sname	gpa
001	John	3.4
002	Bob	1.3

$Students \times People$

ssn	pname	address	sid	sname	gpa
1234545	John	216 Rosse	001	John	3.4
5423341	Bob	217 Rosse	001	John	3.4
1234545	John	216 Rosse	002	Bob	1.3
5423341	Bob	216 Rosse	002	Bob	1.3

4. Renaming (ρ)

- Changes the schema, not the instance
- A 'special' operator- neither basic nor derived
- Notation: $\rho_{B1,\ldots,Bn}(R)$

- Note: this is <u>shorthand</u> for the proper form (since <u>names, not order</u> matters!):
  - $\rho_{A1 \to B1,\ldots,An \to Bn}(R)$

```
Students(sid,sname,gpa)
```

SQL:

```
SELECT
  sid AS studId,
  sname AS name,
  gpa AS gradePtAvg
FROM Students;
```

RA:

$\rho_{studId,name,gradePtAvg}(Students)$

We care about this operator *because* we are working in a *named perspective*

Another example:

Students

sid	sname	gpa
001	John	3.4
002	Bob	1.3

$\rho_{studId,name,gradePtAvg}(Students)$

Students

studId	name	gradePtAvg
001	John	3.4
002	Bob	1.3

5. Natural Join (⋈)

- Notation: $R_1 \bowtie R_2$
- Joins R_1 and R_2 on equality of all shared attributes
  - If R_1 has attribute set A, and R_2 has attribute set B, and they share attributes $A \cap B = C$, can also be written: $R_1 \bowtie_C R_2$
- Our first example of a derived RA operator:
  - Meaning: $R_1 \bowtie R_2 = \Pi_{A \cup B}(\sigma_{C=D}(\rho_{C \to D}(R_1) \times R_2))$
  - Where:
    - The rename $\rho_{C \to D}$ renames the shared attributes in one of the relations
    - The selection $\sigma_{C=D}$ checks equality of the shared attributes
    - The projection $\Pi_{A \cup B}$ eliminates the duplicate common attributes

```
Students(sid,name,gpa)
People(ssn,name,address)
```

SQL:

```
SELECT DISTINCT
  ssid, S.name, gpa,
  ssn, address
FROM
  Students S,
  People P
WHERE S.name = P.name;
```

RA:

$$Students \bowtie People$$

Another example:

Students S

sid	S.name	gpa
001	John	3.4
002	Bob	1.3

$\bowtie$

People P

ssn	P.name	address
1234545	John	216 Rosse
5423341	Bob	217 Rosse

$$Students \bowtie People$$

sid	S.name	gpa	ssn	address
001	John	3.4	1234545	216 Rosse
002	Bob	1.3	5423341	216 Rosse

Natural Join practice

- Given schemas R(A, B, C, D), S(A, C, E), what is the schema of R ⋈ S ?

- Given R(A, B, C), S(D, E), what is R ⋈ S ?

- Given R(A, B), S(A, B), what is R ⋈ S ?

Example: Converting SFW Query -> RA

```
Students(sid,sname,gpa)
People(ssn,sname,address)
```

```
SELECT DISTINCT
   gpa,
   address
FROM Students S,
      People P
WHERE gpa > 3.5 AND
    sname = pname;
```

$$\Pi_{gpa,address}(\sigma_{gpa>3.5}(S \bowtie P))$$

How do we represent this query in RA?

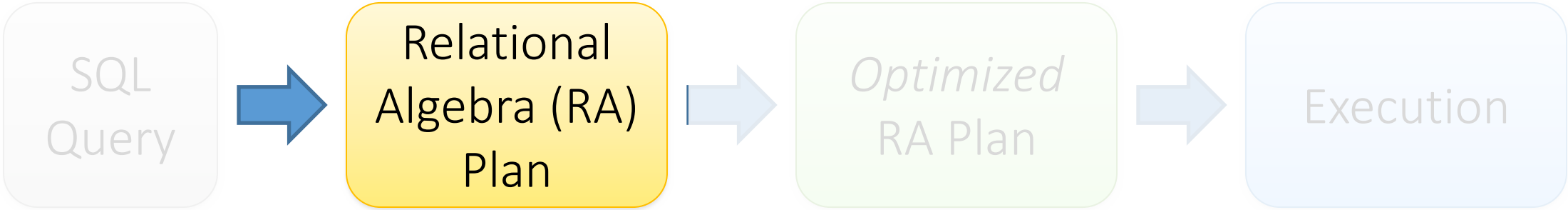
Logical Equivalece of RA Plans

- Given relations R(A,B) and S(B,C):
  - Here, projection & selection commute:
    - $\sigma_{A=5}(\Pi_A(R)) = \Pi_A(\sigma_{A=5}(R))$
  - What about here?
    - $\sigma_{A=5}(\Pi_B(R)) \,?= \Pi_B(\sigma_{A=5}(R))$

We'll look at this in more depth in query optimization…

RDBMS Architecture

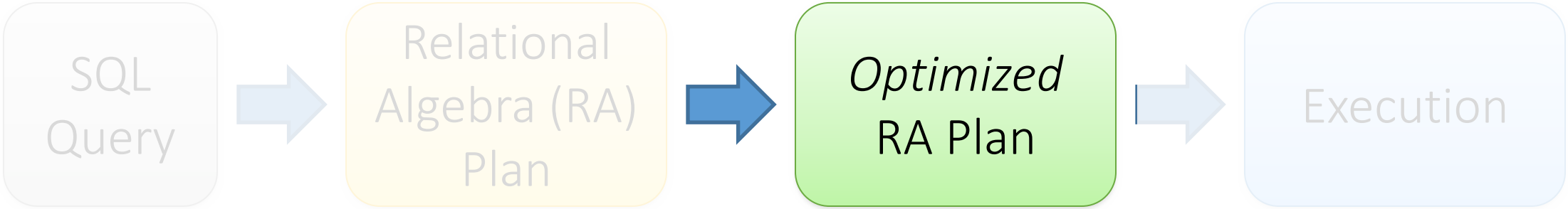
- How does a SQL engine work ?



We saw how we can transform declarative SQL queries into precise, compositional RA plans

RDBMS Architecture

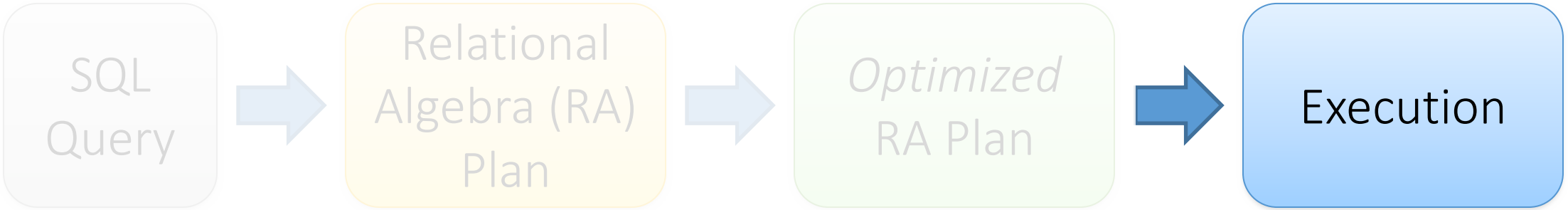
- How does a SQL engine work ?



We'll look at how to then optimize these plans

RDBMS Architecture

- How is the RA “plan” executed?



We have already seen how to execute a few basic operators!

RA Plan Execution

- Natural Join / Join:
  - We saw how to use memory & IO cost considerations to pick the correct algorithm to execute a join with BNLJ or SMJ (we skipped HJ)
- Selection:
  - We saw how to use indexes to aid selection
  - Can always fall back on scan / binary search as well
- Projection:
  - The main operation here is finding distinct values of the project tuples; we briefly discussed how to do this with sorting (we skipped hashing)

We already know how to execute all the basic operators!

3. Advanced Relational Algebra (very brief)

What we will briefly cover next

- Set Operations in RA
- Extensions & Limitations

Relational Algebra (RA)

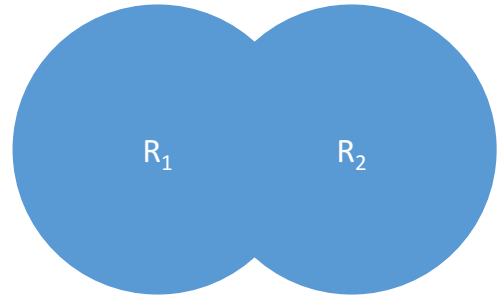
- Five basic operators:
  1. Selection: σ
  2. Projection: Π
  3. Cartesian Product: $\times$
  4. Union: $\cup$
  5. Difference: -

We'll look at these

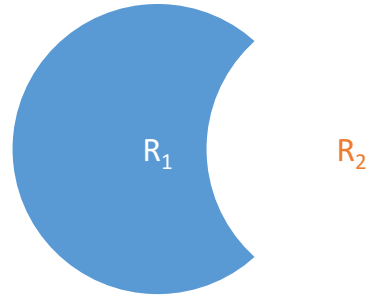
- Derived or auxiliary operators:
  - Intersection, complement
  - Joins (natural,equi-join, theta join, semi-join)
  - Renaming: Π
  - Division

1. Union (∪) and 2. Difference (–)

- R1 ∪ R2
- Example:
  - ActiveEmployees ∪ RetiredEmployees

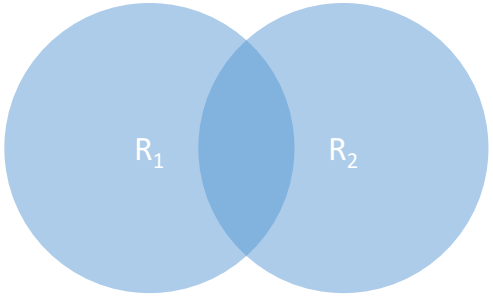


- R1 – R2
- Example:
  - AllEmployees -- RetiredEmployees

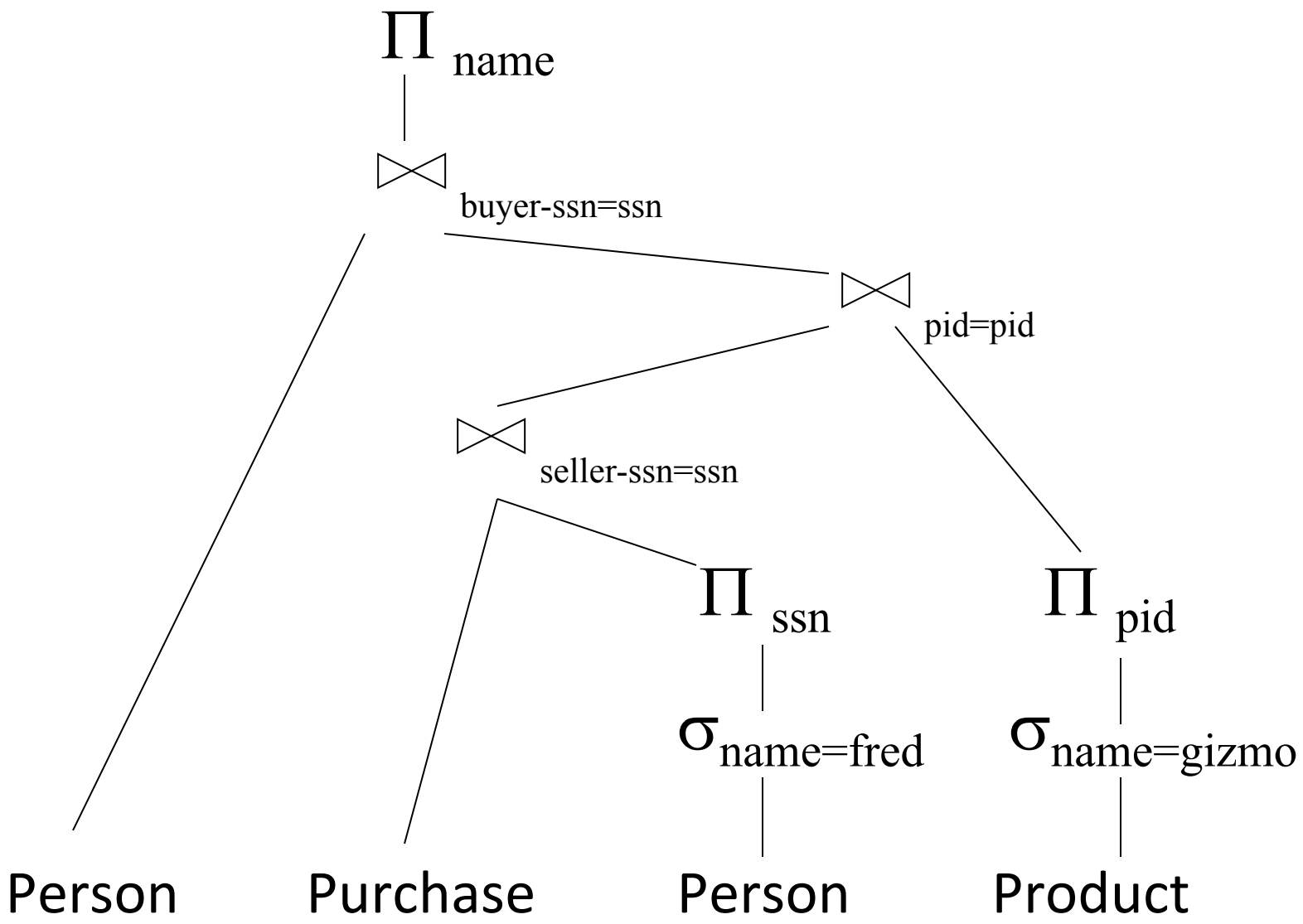


What about Intersection (∩) ?

- It is a derived operator
- R1 ∩ R2 = R1 – (R1 – R2)
- Also expressed as a join!
- Example
  - UnionizedEmployees ∩ RetiredEmployees



RA Expressions Can Get Complex!



Operations on Multisets

- All RA operations need to be defined carefully on bags
  - $\sigma_C(R)$: preserve the number of occurrences
  - $\Pi_A(R)$: no duplicate elimination
  - Cross-product, join: no duplicate elimination

This is important- relational engines work on multisets, not sets!

RA has Limitations !

- Cannot compute “transitive closure”

Name1	Name2	Relationship
Fred	Mary	Father
Mary	Joe	Cousin
Mary	Bill	Spouse
Nancy	Lou	Sister

- Find all direct and indirect relatives of Fred
- Cannot be expressed in RA !
  - Need to write C program, use a graph engine, or modern SQL…

Activity-45.ipynb

as part of HW6

L22: Query Optimization

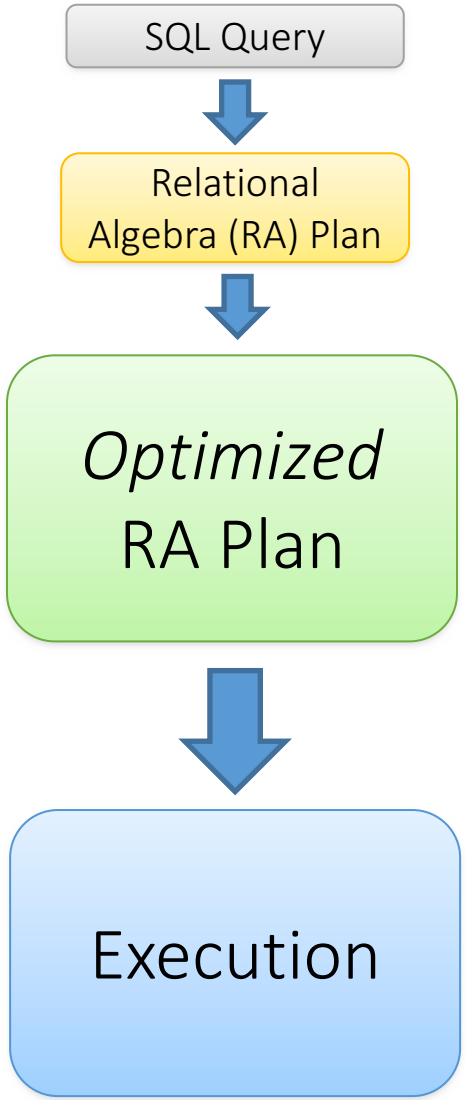
CS3200 Database design (sp18 s2)

https://course.ccs.neu.edu/cs3200sp18s2/

4/5/2018

Logical vs. Physical Optimization

- Logical optimization:
  - Find equivalent plans that are more efficient
  - Intuition: Minimize # of tuples at each step by changing the order of RA operators
- Physical optimization:
  - Find algorithm with lowest IO cost to execute our plan
  - Intuition: Calculate based on physical parameters (buffer size, etc.) and estimates of data size (histograms)
- We only discuss Logical optimization today

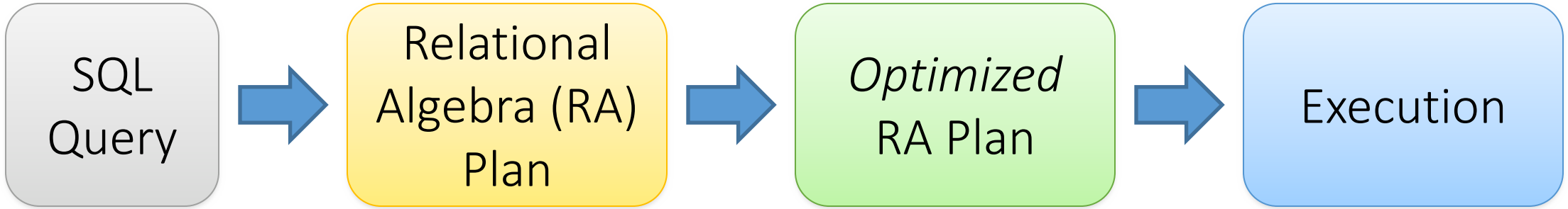


1. Logical Optimization

1) Optimization of RA Plans
2) ACTIVITY: RA Plan Optimization

RDBMS Architecture

- How does a SQL engine work ?



Declarative query (from user)

Translate to relational algebra expresson

Find logically equivalent- but more efficient- RA expression

Execute each operator of the optimized plan!

RDBMS Architecture

- How does a SQL engine work ?

Relational Algebra allows us to translate declarative (SQL) queries into precise and optimizable expressions!

Relational Algebra (RA)

- Five basic operators:
  1. Selection: σ
  2. Projection: Π
  3. Cartesian Product: $\times$
  4. Union: $\cup$
  5. Difference: -
- Derived or auxiliary operators:
  - Intersection, complement
  - Joins (natural, equi-join, theta join, semi-join)
  - Renaming: ρ
  - Division

We'll look at these first!

And also at one example of a derived operator (natural join) and a special operator (renaming)

Recall: Converting SFW Query -> RA

```
Students(sid,sname,gpa)
People(ssn,sname,address)
```

```
SELECT DISTINCT
   gpa,
   address
FROM Students S,
      People P
WHERE gpa > 3.5 AND
    sname = pname;
```

$$\Pi_{gpa,address}(\sigma_{gpa>3.5}(S \bowtie P))$$

How do we represent this query in RA?

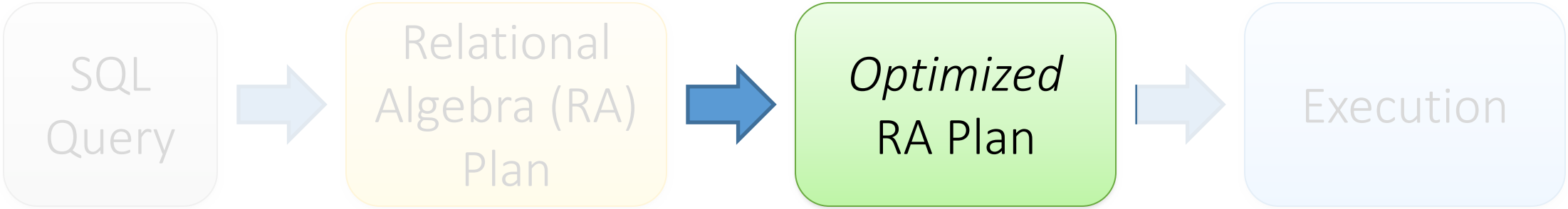
Recall: Logical Equivalence of RA Plans

- Given relations R(A,B) and S(B,C):
  - Here, projection & selection commute:
    - $\sigma_{A=5}(\Pi_A(R)) = \Pi_A(\sigma_{A=5}(R))$
  - What about here?
    - $\sigma_{A=5}(\Pi_B(R)) \stackrel{?}{=} \Pi_B(\sigma_{A=5}(R))$

We'll look at this in more depth later in the lecture…

RDBMS Architecture

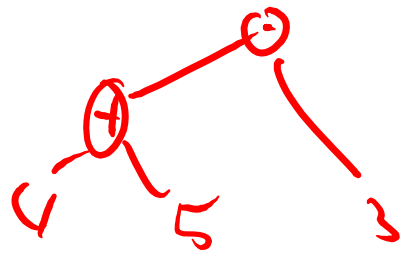
- How does a SQL engine work ?



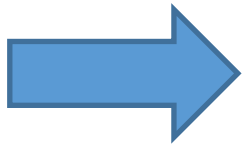
We'll look at how to then optimize these plans now

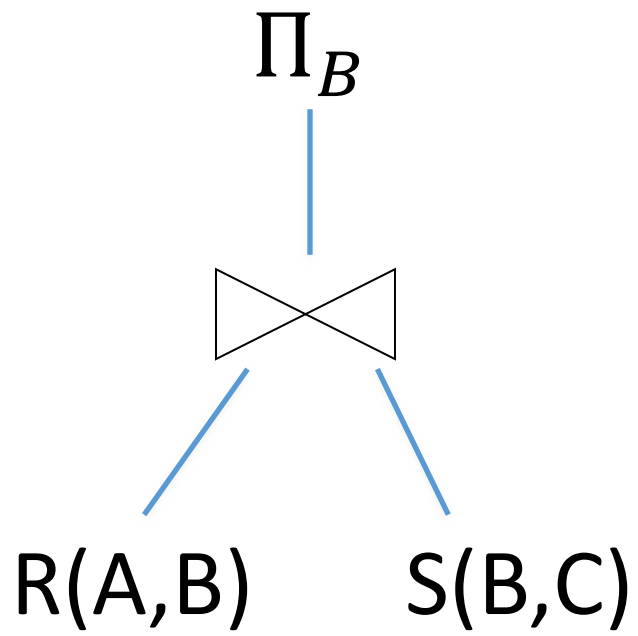
Note: We can visualize the plan as a tree

$3 \cdot (4 + 5)$



$$\Pi_B(R(A,B) \bowtie S(B,C))$$



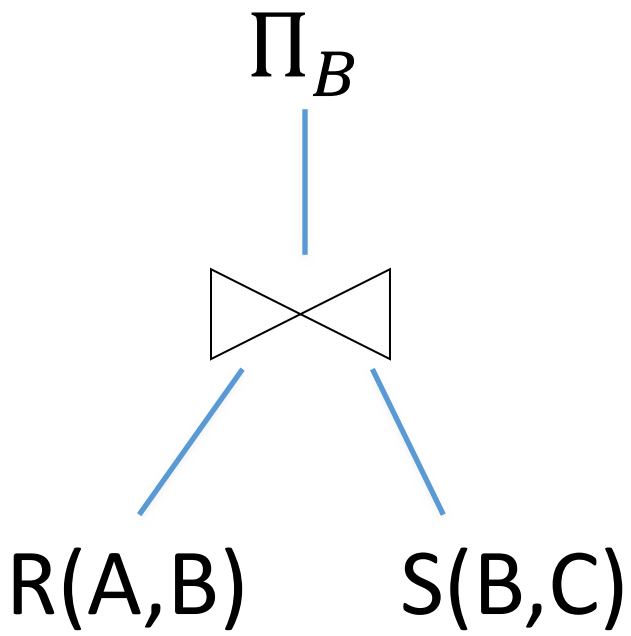


Bottom-up tree traversal = order of operation execution!

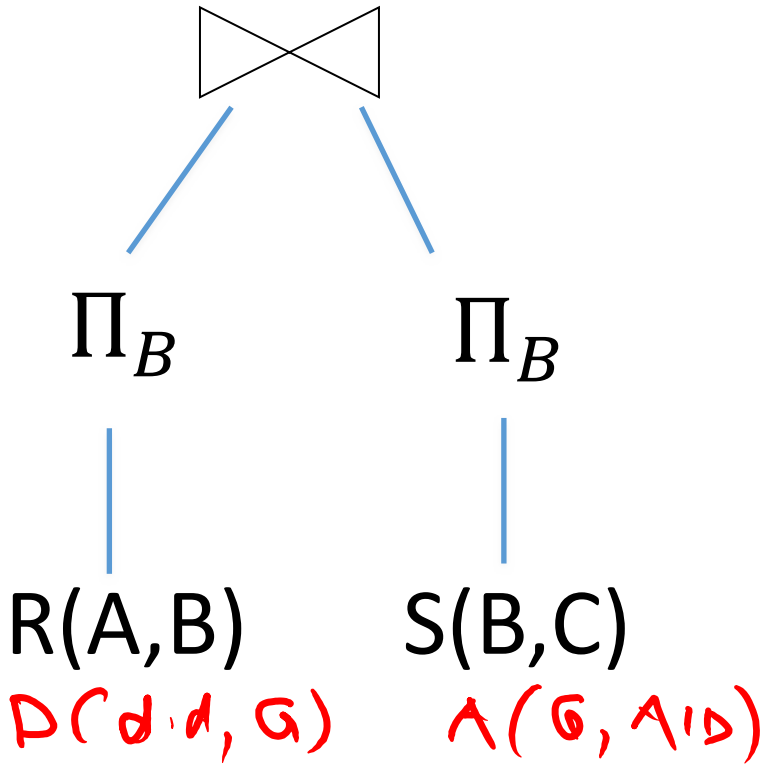
A simple plan

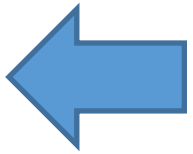
What SQL query does this correspond to?

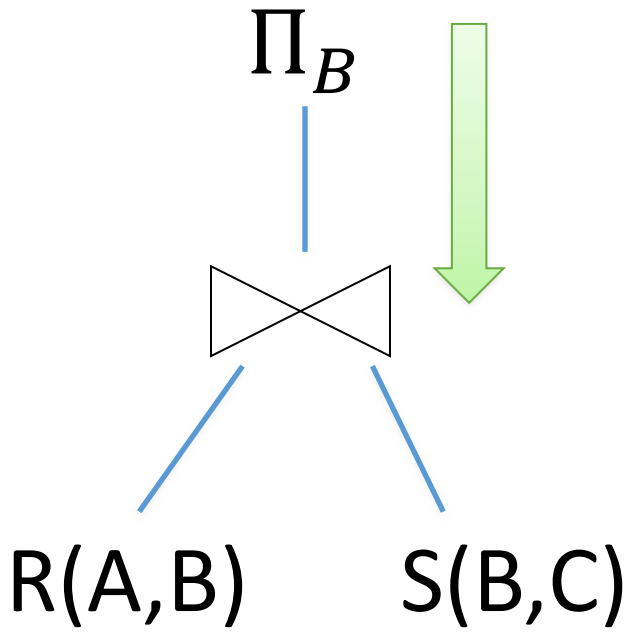
Are there any logically equivalent RA expressions?



“Pushing down” projection







Why might we prefer this plan?

Takeaways

- This process is called logical optimization
- Many equivalent plans used to search for “good plans”
- Relational algebra is an important abstraction.

RA commutators

- The basic commutators:
  - Push projection through (1) selection, (2) join
  - Push selection through (3) selection, (4) projection, (5) join
  - Also: Joins can be re-ordered!

- Note that this is not an exhaustive set of operations

This simple set of tools allows us to greatly improve the execution time of queries by optimizing RA plans!

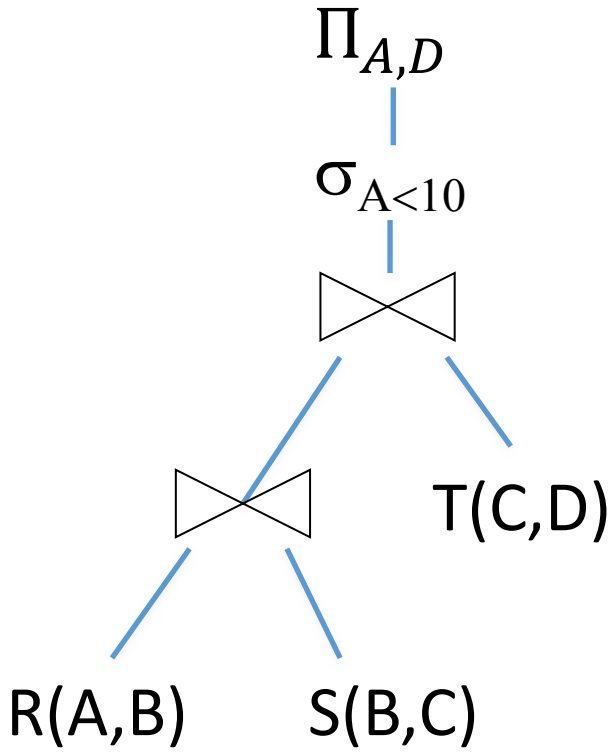
Optimizing the SFW RA Plan

Translating to RA

R(A,B) S(B,C) T(C,D)

```
SELECT R.A,S.D
FROM R,S,T
WHERE R.B = S.B
  AND S.C = T.C
  AND R.A < 10;
```

$$\Pi_{A,D}(\sigma_{A<10}(T \bowtie (R \bowtie S)))$$



Logical Optimization

- Heuristically, we want selections and projections to occur as early as possible in the plan
  - Terminology: "push down selections" and "pushing down projections."

- Intuition: We will have fewer tuples in a plan.
  - Could fail if the selection condition is very expensive (say runs some image processing algorithm).
  - Projection could be a waste of effort, but more rarely.

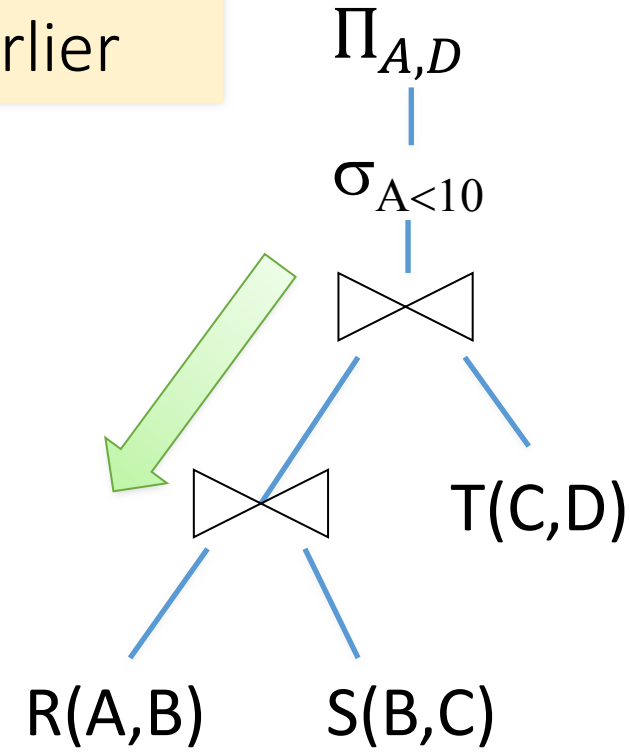
Optimizing RA Plan

```
R(A,B)  S(B,C)  T(C,D)
```

```
SELECT R.A,S.D
FROM R,S,T
WHERE R.B = S.B
  AND S.C = T.C
  AND R.A < 10;
```

Push down selection on A so it occurs earlier

$$\Pi_{A,D}(\sigma_{A<10}(T \bowtie (R \bowtie S)))$$



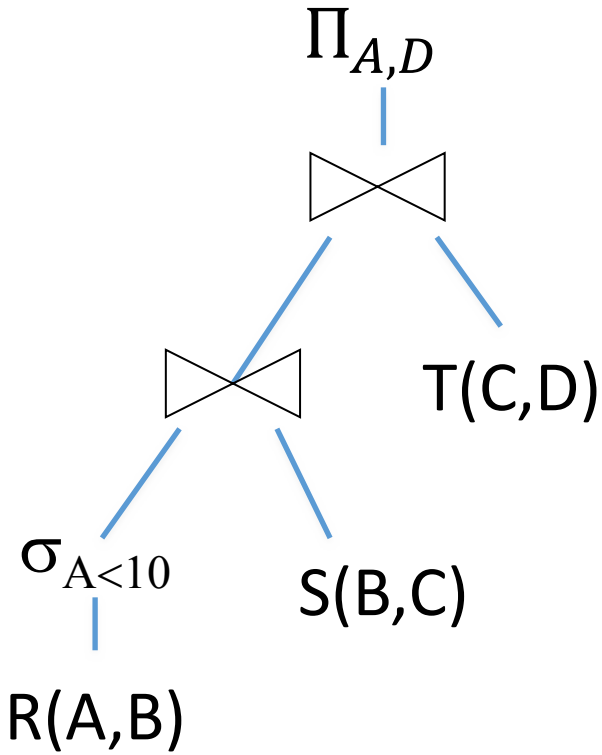
Optimizing RA Plan

```
R(A,B)  S(B,C)  T(C,D)
```

```
SELECT R.A,S.D
FROM R,S,T
WHERE R.B = S.B
  AND S.C = T.C
  AND R.A < 10;
```

Push down selection on A so it occurs earlier

$$\Pi_{A,D}\left(T \bowtie \left(\sigma_{A<10}(R) \bowtie S\right)\right)$$



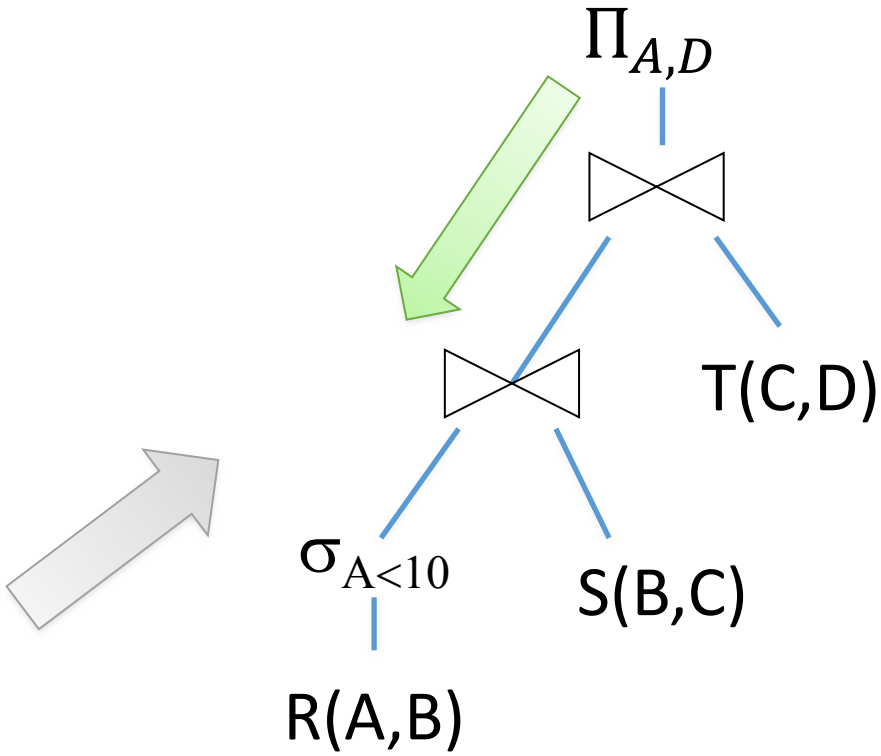
Optimizing RA Plan

```
R(A,B)  S(B,C)  T(C,D)
```

```
SELECT R.A,S.D
FROM R,S,T
WHERE R.B = S.B
  AND S.C = T.C
  AND R.A < 10;
```

Push down projection so it occurs earlier

$$\Pi_{A,D}\left(T \bowtie \left(\sigma_{A<10}(R) \bowtie S\right)\right)$$



Optimizing RA Plan

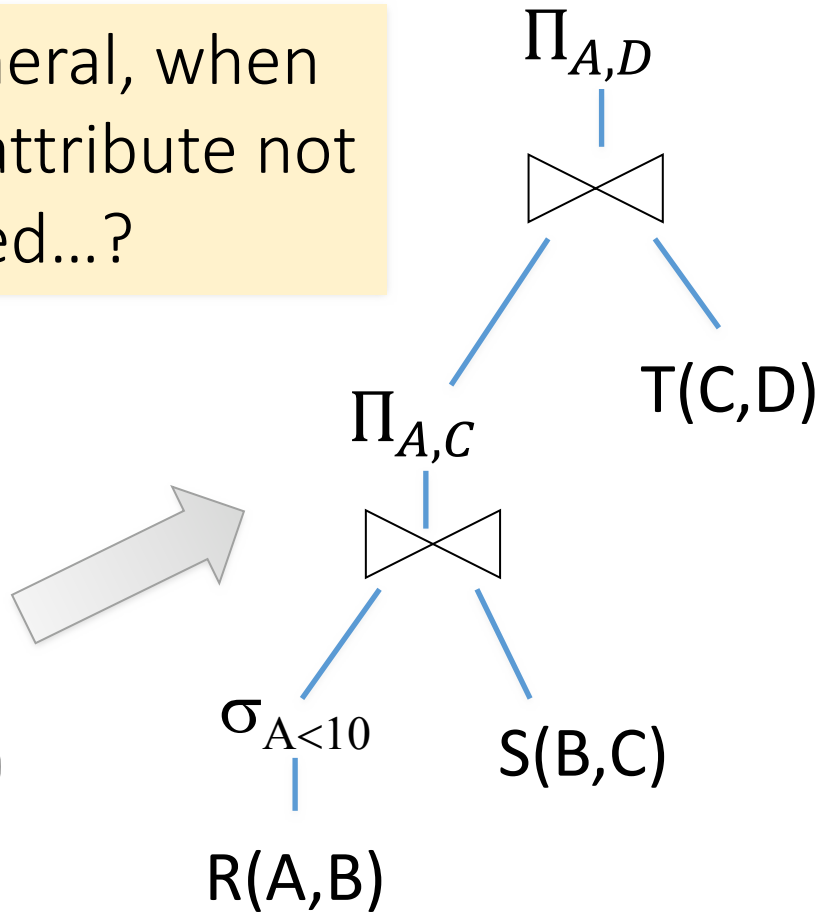
```
R(A,B)  S(B,C)  T(C,D)
```

```
SELECT R.A,S.D
FROM R,S,T
WHERE R.B = S.B
  AND S.C = T.C
  AND R.A < 10;
```

We eliminate B earlier!

In general, when is an attribute not needed…?

$$\Pi_{A,D}\left(T \bowtie \Pi_{A,C}(\sigma_{A<10}(R) \bowtie S)\right)$$



Activity-47.ipynb

as part of HW6