L22: Relational Algebra

CS3200 Database design (fa18 s2)

https://northeastern-datalab.github.io/cs3200/

Version 11/29/2018

Announcements!

- Exam 3 feedback
  - Question: what if we changed to paper only?
    - Plus: no computer setup problems, more controlled environment
    - Minus: not realistic (syntax errors ...)
- Please bring your laptops next Monday for course evaluations
- Today
  - Relational Algebra
  - Optimization

Schedule

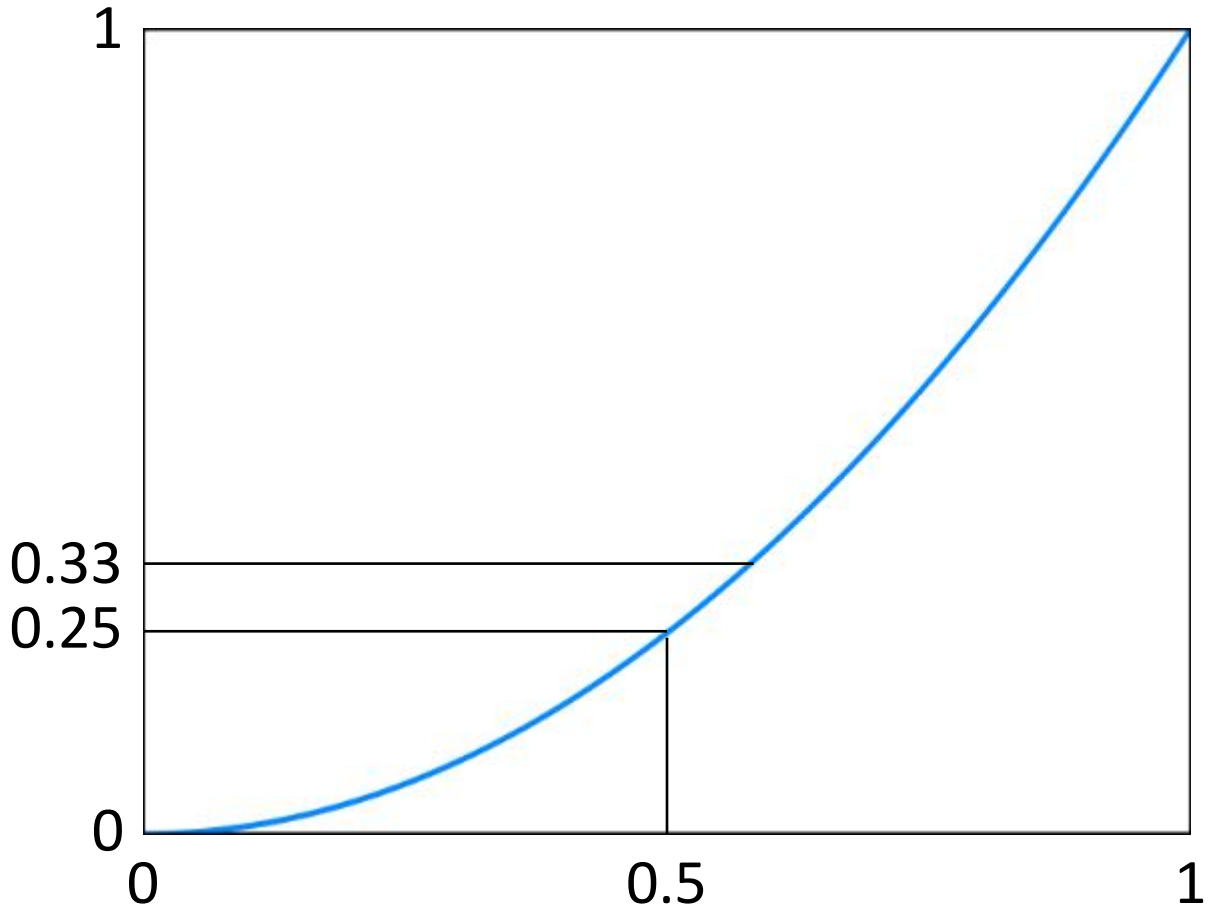
		Data models and Query Processing		
21	M Nov 26	Data Models, Relational Algebra	**Stonebraker, Hellerstein**	
22	R Nov 29	Relational Algebra & Query Optimization	**Gehrke, Ramakrishnan**: Ch 4	HW7
23	M Dec 3	Course Evaluation, Class Review		HW8, HW10, Q11 Optional PPTX (all due by Dec 5)
	R Dec 6	No class: Reading day		
	T Dec 11	**Exam 3**: 8am-10am, location: Behrakis Health Sciences Cntr 310		

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?
    - $\sigma_{A=5}(\Pi_B(R)) \,?= \Pi_B(\sigma_{A=5}(R))$

Commuting functions

- Do functions commute with taking the expectation?
  - E[f(x)] = f(E[x]) ?
- Only for linear functions
  - Thus f(x)=ax + b
  - E[ax+b] = a E[x] + b
- Counter example $f(x) = x^2$
  - Assume $0 \leq x \leq 1$
  - f(E[x]) = f(0.5) = 0.25
  - $E[f(x)] = \frac{\int_0^1 f(x)}{1-0} = \frac{x^3}{3}\Big|_0^1 = 0.33$



Relational Algebra (RA)

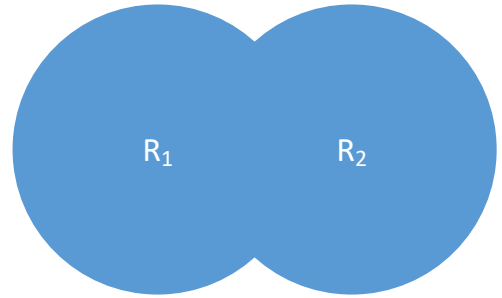
- Five basic operators:
  1. Selection: σ
  2. Projection: Π
  3. Cartesian Product: $\times$
  4. Union: $\cup$
  5. Difference: -
- Auxiliary operators (sometimes counted as basic):
  - Renaming: Π
- Derived or Intersection, complement
  - Joins (natural,equi-join, theta join, semi-join)
  - Division

Implied Operators

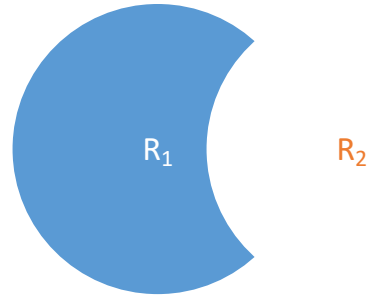
- Derived relational operators
  - Not among the 5 basic operators (sometimes 6 if renaming counted)
  - Can be expressed in RA (implied)
  - Very common in practice
- Enhancing the available operator set with the implied operators is a good idea!
  - Easier to write queries
  - Easier to understand/maintain queries
  - Easier for DBMS to apply specialized optimizations

1. Union (∪) and 2. Difference (–)

- R1 ∪ R2
- Example:
  - ActiveEmployees ∪ RetiredEmployees

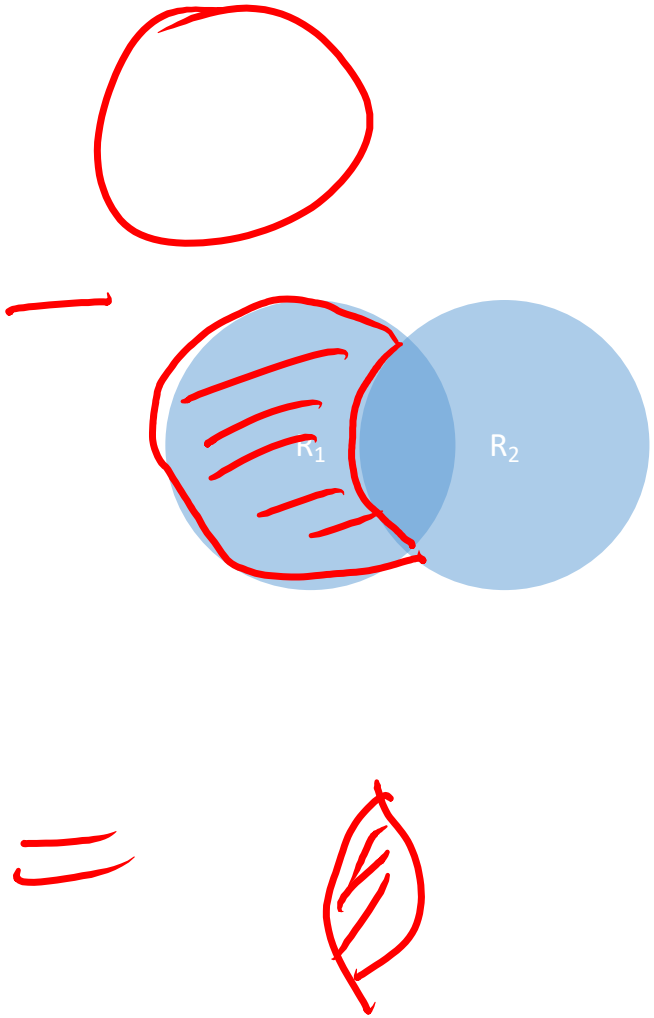


- R1 – R2
- Example:
  - AllEmployees -- RetiredEmployees



What about Intersection (∩) ?

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



Theta Join ($\bowtie_\theta$)

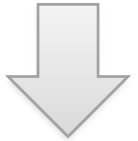
- A join that involves a predicate
- $R1 \bowtie_\theta R2 = \sigma_\theta (R1 \times R2)$
- Here θ can be any condition

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

SQL:

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

Note that natural join is a theta join + a projection.



RA:

$$Students \bowtie_\theta People$$

Equi-join ($\bowtie_{A=B}$)

- A theta join where q is an equality
- $R1 \bowtie_{A=B} R2 = \sigma_{A=B} (R1 \times R2)$
- Example:
  - Employee $\bowtie_{SSN=SSN}$ Dependents

Most common join in practice!

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

SQL:

```
SELECT *
FROM
  Students S,
  People P
WHERE sname = pname;
```

RA:

$$S \bowtie_{sname=pname} P$$

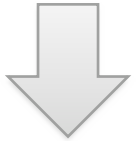
Semijoin (⋉)

- $R \ltimes S = \Pi_{A1,\ldots,An} (R \bowtie S)$
  - where $A_1, \ldots, A_n$ are the attributes in R
- Intuition:
  - remove "dangling tuples"
  - Whiteboard: how to specify in SQL?
    - 3 variants
- Example:
  - Employee ⋉ Dependents

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

SQL:

```
SELECT DISTINCT
  sid,sname,gpa
FROM
  Students,People
WHERE
  sname = pname;
```

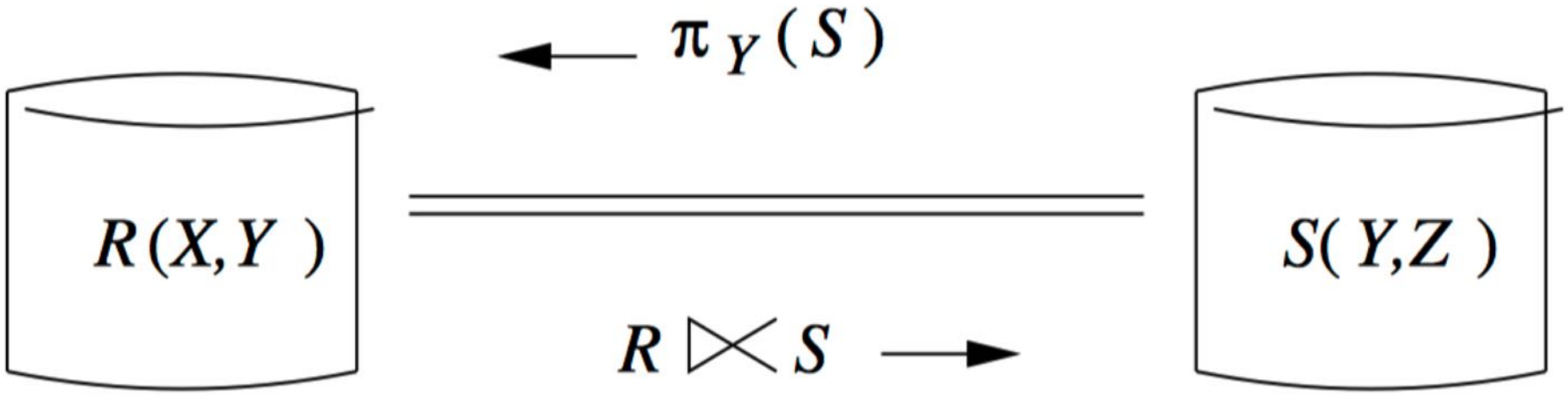


RA:

$$Students \ltimes People$$

Semijoins in Distributed Databases

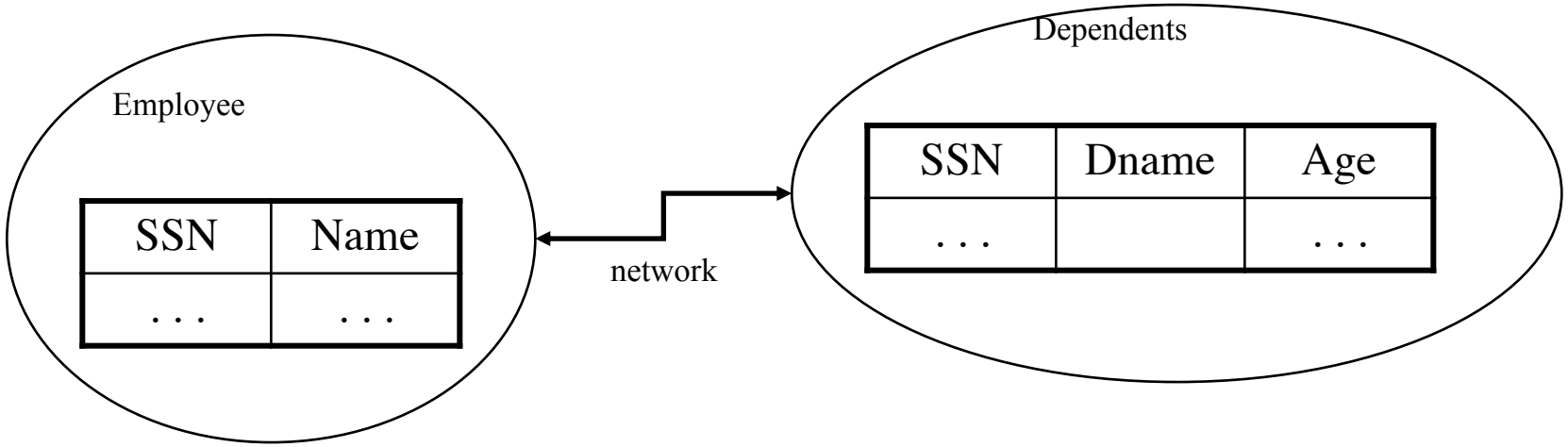
- Semijoins are often used to compute natural joins in distributed databases



Send less data to reduce network bandwidth!

Semijoins in Distributed Databases

- Semijoins are often used to compute natural joins in distributed databases



Send less data to reduce network bandwidth!

$$\text{Employee} \bowtie_{ssn=ssn} (\sigma_{age>71} (\text{Dependents}))$$

$T = \Pi_{SSN}\, \sigma_{age>71}$ (Dependents)

$R = \text{Employee} \ltimes T$

$\text{Answer} = R \bowtie \text{Dependents}$

Division

- Consider two relations R(X,Y) and S(Y)
  - Here, X and Y are tuples of attributes

- R ÷ S is the relation T(X) that contains all the Xs that occur with *every* Y in S

Formal Definition

- **Legal input:** (R,S) such that R has all the attributes of S
- R÷S is the relation T with:
  - The header of R, with all attributes of S removed
  - Tuple set {t[**X**] | t[**X**,**Y**]∈R for every s[**Y**]∈S}
    - *This is an abuse of notation, since the attributes in* X *need not necessarily come before those of* Y

Questions

sid	student	course
861	Alma	DB
861	Alma	PL
753	Amir	DB
753	Amir	AI
955	Ahuva	PL
955	Ahuva	DB
955	Ahuva	AI

÷

course
AI

= ?

÷

course
DB
PL
AI

= ?

$(R \times S) \div S = ?$

$(R \times S) \div R = ?$

Q: If R has 1000 tuples and S has 100 tuples, how many tuples can be in R÷S?

Q: If R has 1000 tuples and S has 1001 tuples, how many tuples can be in R÷S?

Studies

sid	student	course
861	Alma	DB
861	Alma	PL
753	Amir	DB
753	Amir	AI
955	Ahuva	PL
955	Ahuva	DB
955	Ahuva	AI

CourseType

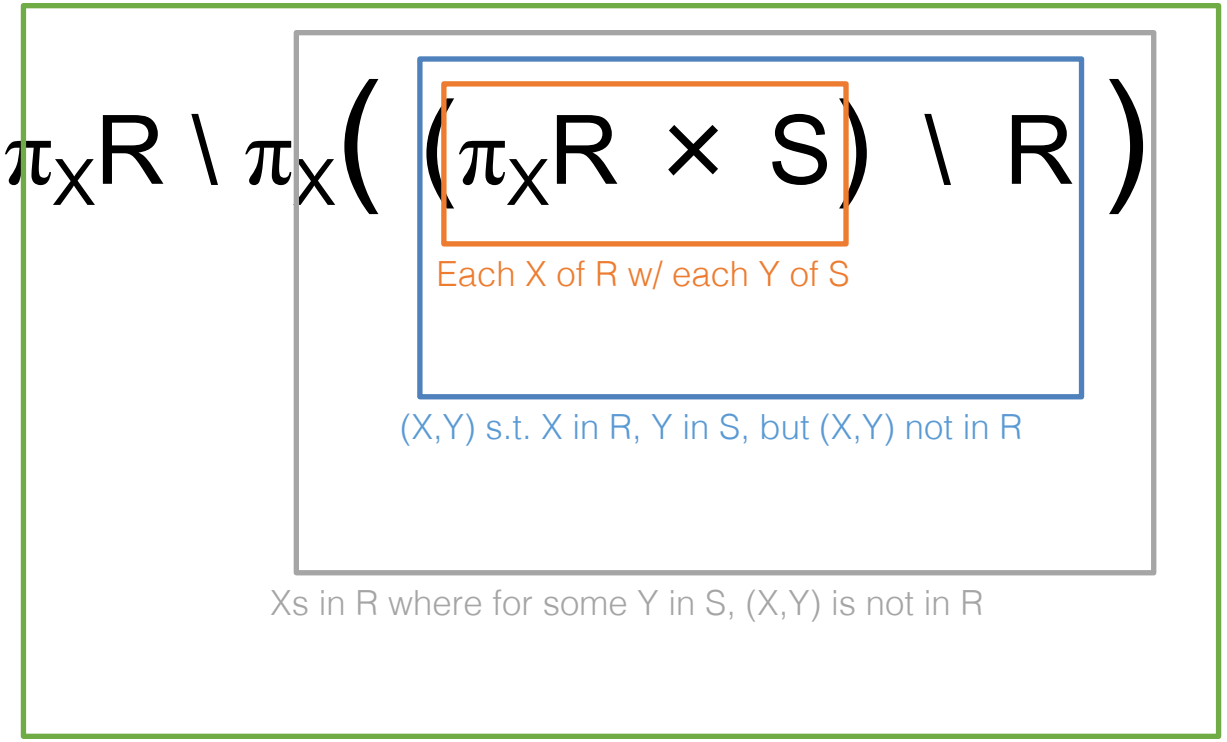
course	type
DB	core
PL	core
AI	elective
DC	elective

Who took all core courses?

$$\text{Studies} \div \pi_{\text{course}}\sigma_{\text{type='core'}}\text{CourseType}$$

R÷S in Primitive RA

R(X,Y) S(Y)



Multisets

Recall that SQL uses Multisets

Multiset X

Tuple
(1, a)
(1, a)
(1, b)
(2, c)
(2, c)
(2, c)
(1, d)
(1, d)

Equivalent Representations of a **Multiset**

$\boldsymbol{\lambda(X)}$= *"Count of tuple in X" (Items not listed have implicit count 0)*

Multiset X

Tuple	$\boldsymbol{\lambda(X)}$
(1, a)	2
(1, b)	1
(2, c)	3
(1, d)	2

Note: In a set all counts are {0,1}.

Generalizing Set Operations to Multiset Operations

Multiset X

Tuple	$\lambda(X)$
(1, a)	2
(1, b)	0
(2, c)	3
(1, d)	0

$\cap$

Multiset Y

Tuple	$\lambda(Y)$
(1, a)	5
(1, b)	1
(2, c)	2
(1, d)	2

$=$

Multiset Z

Tuple	$\lambda(Z)$
(1, a)	2
(1, b)	0
(2, c)	2
(1, d)	0

$$\lambda(Z) = min(\lambda(X), \lambda(Y))$$

For sets, this is **intersection**

Generalizing Set Operations to Multiset Operations

Multiset X

Tuple	$\lambda(X)$
(1, a)	2
(1, b)	0
(2, c)	3
(1, d)	0

$\cup$

Multiset Y

Tuple	$\lambda(Y)$
(1, a)	5
(1, b)	1
(2, c)	2
(1, d)	2

$=$

Multiset Z

Tuple	$\lambda(Z)$
(1, a)	7
(1, b)	1
(2, c)	5
(1, d)	2

$$\lambda(Z) = \lambda(X) + \lambda(Y)$$

For sets,
this is **union**

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!

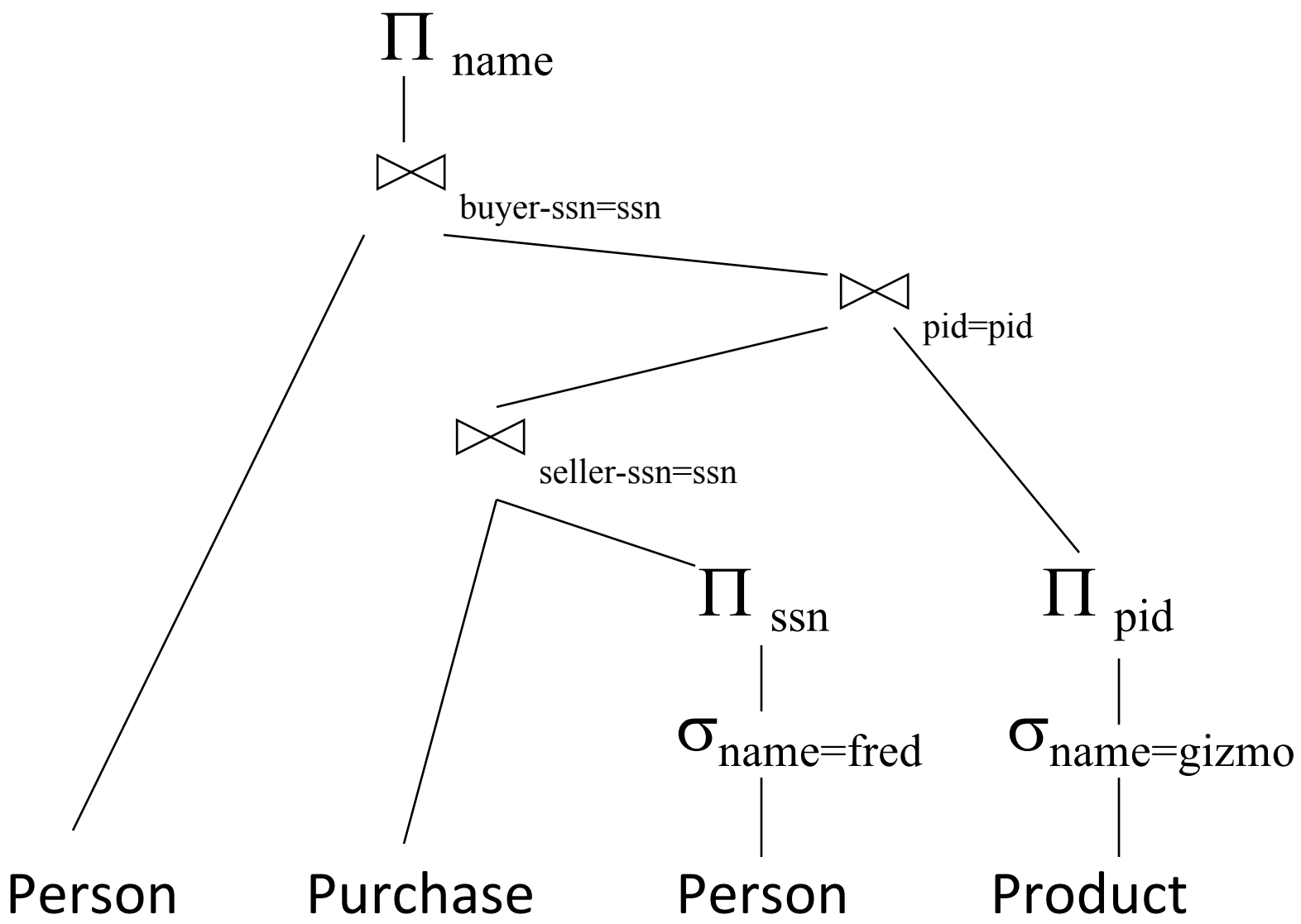
RA has Limitations !

- 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
- Aggregates
- Cannot be expressed in RA !
  - Need to write C program, use a graph engine, or modern SQL…

RA Expressions Can Get Complex!



Activity-55.ipynb

as part of HW8

Parentheses Convention

- We have defined 3 unary operators and 3 binary operators
- It is acceptable to omit the parentheses from o(R) when o is unary
  - Then, unary operators take precedence over binary ones
- Example:

$$(\sigma_{course='DB'}(Course)) \times (\rho_{cid/cid1}(Studies))$$

becomes

$$\sigma_{course='DB'}Course \times \rho_{cid/cid1}Studies$$

Composition Example (courtesy Benny Kimelfeld)

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}\text{Student} \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}\text{Course} \times \rho_{cid/cid1}\text{Studies}))))$$

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

Composition Example (courtesy Benny Kimelfeld)

Names of students who study DB:

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}\text{Student} \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}\text{Course} \times \rho_{cid/cid1}\text{Studies}))))$$

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}Student \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}Course \times \rho_{cid/cid1}Studies))))$$

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}Student \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}Course \times \rho_{cid/cid1}Studies))))$$

sid	cid1
861	23
861	45
753	45
955	76

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}Student \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}Course \times \rho_{cid/cid1}Studies))))$$

cid	topic
45	DB

sid	cid1
861	23
861	45
753	45
955	76

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}Student \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}Course \times \rho_{cid/cid1}Studies))))$$

cid	topic	sid	cid1
45	DB	861	23
45	DB	861	45
45	DB	753	45
45	DB	955	76

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}Student \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}Course \times \rho_{cid/cid1}Studies))))$$

cid	topic	sid	cid1
45	DB	861	45
45	DB	753	45

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}\text{Student} \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}\text{Course} \times \rho_{cid/cid1}\text{Studies}))))$$

cid	sid
45	861
45	753

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}Student \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}Course \times \rho_{cid/cid1}Studies))))$$

sid1	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

cid	sid
45	861
45	753

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}Student \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}Course \times \rho_{cid/cid1}Studies))))$$

sid1	name	year	cid	sid
861	Alma	2	45	861
753	Amir	1	45	861
955	Ahuva	2	45	861
861	Alma	2	45	753
753	Amir	1	45	753
955	Ahuva	2	45	753

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}Student \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}Course \times \rho_{cid/cid1}Studies))))$$

sid1	name	year	cid	sid
861	Alma	2	45	861
753	Amir	1	45	753

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

$$\pi_{name}(\sigma_{sid=sid1}(\rho_{sid/sid1}Student \times \pi_{sid,cid}(\sigma_{cid=cid1}(\sigma_{topic='DB'}Course \times \rho_{cid/cid1}Studies))))$$

name
Alma
Amir

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45
955	76

Exercise

Student

sid	name	year
861	Alma	2
753	Amir	1
955	Ahuva	2

Course

cid	topic
23	PL
45	DB
76	OS

Studies

sid	cid
861	23
861	45
753	45

Write a query in RA that finds the names of students who get "private lessons"

(i.e., the student takes a course that no one else takes)

L22: Query Optimization

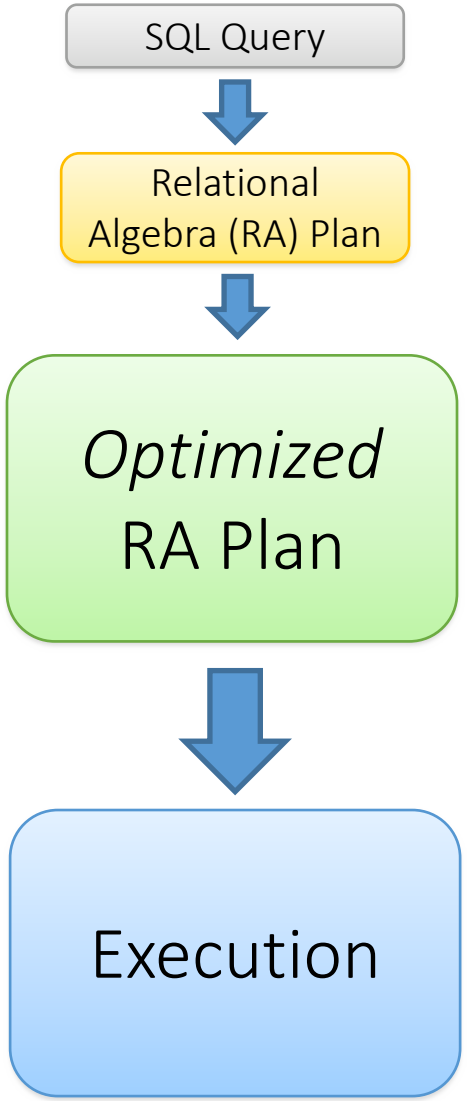
CS3200 Database design (fa18 s2)

https://northeastern-datalab.github.io/cs3200/

Version 11/29/2018

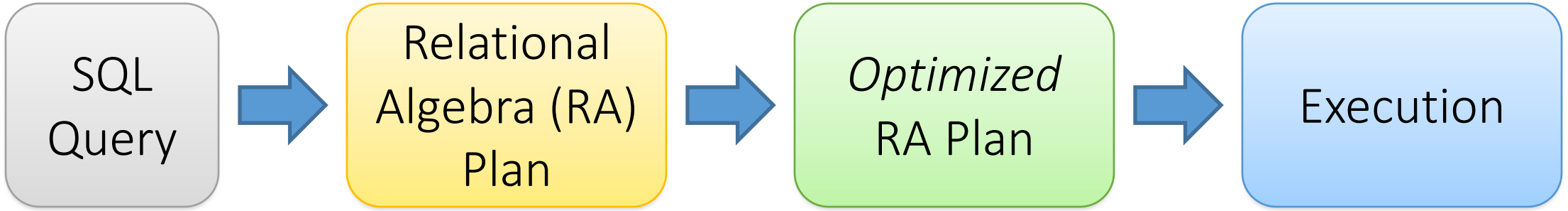
Logical vs. Physical Optimization

- 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



RDBMS Architecture

- How does a SQL engine work ?



Declarative query (from user)

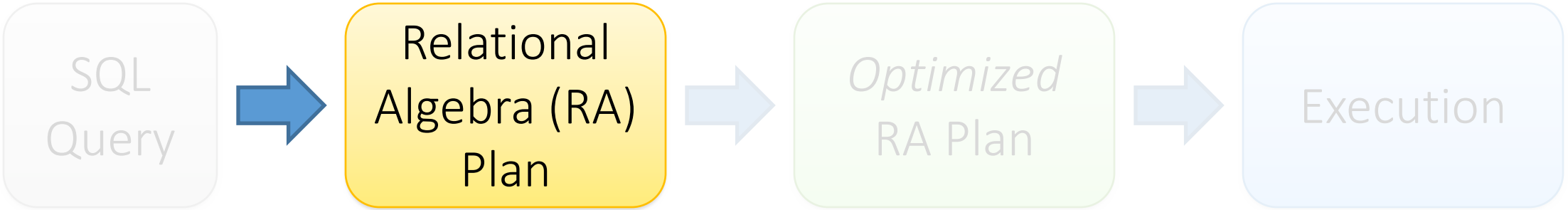
Translate to relational algebra expresson

Find logically equivalent- but more efficient- RA expression

Execute each operator of the optimized plan!

RDBMS Architecture

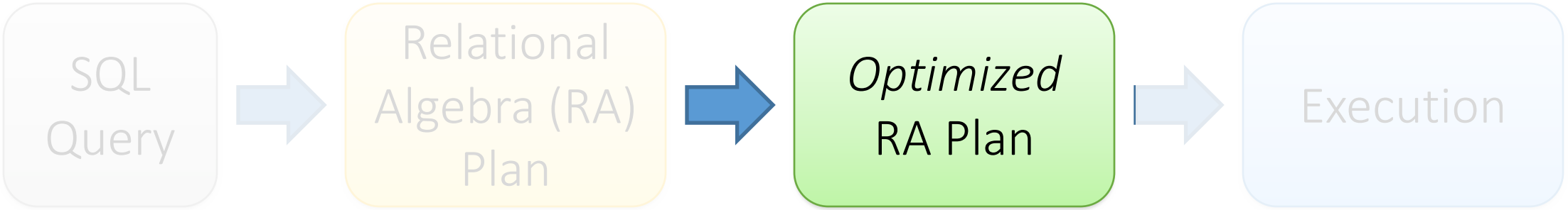
- How does a SQL engine work ?



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

RDBMS Architecture

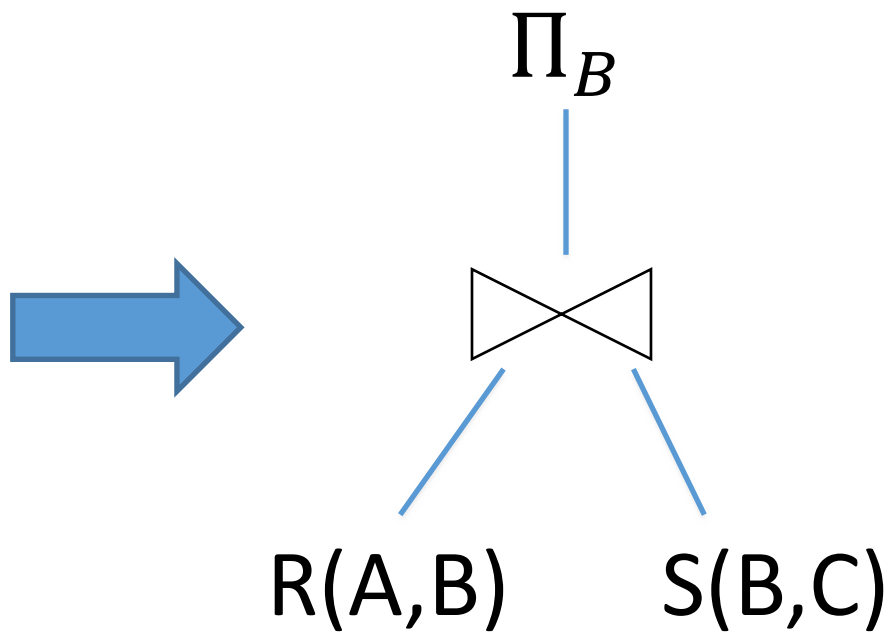
- How does a SQL engine work ?



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

Note: We can visualize the plan as a tree

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

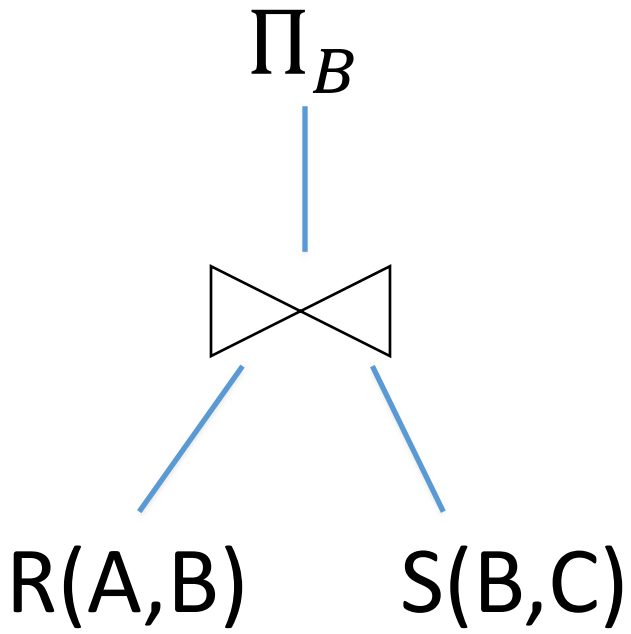


Bottom-up tree traversal = order of operation execution!

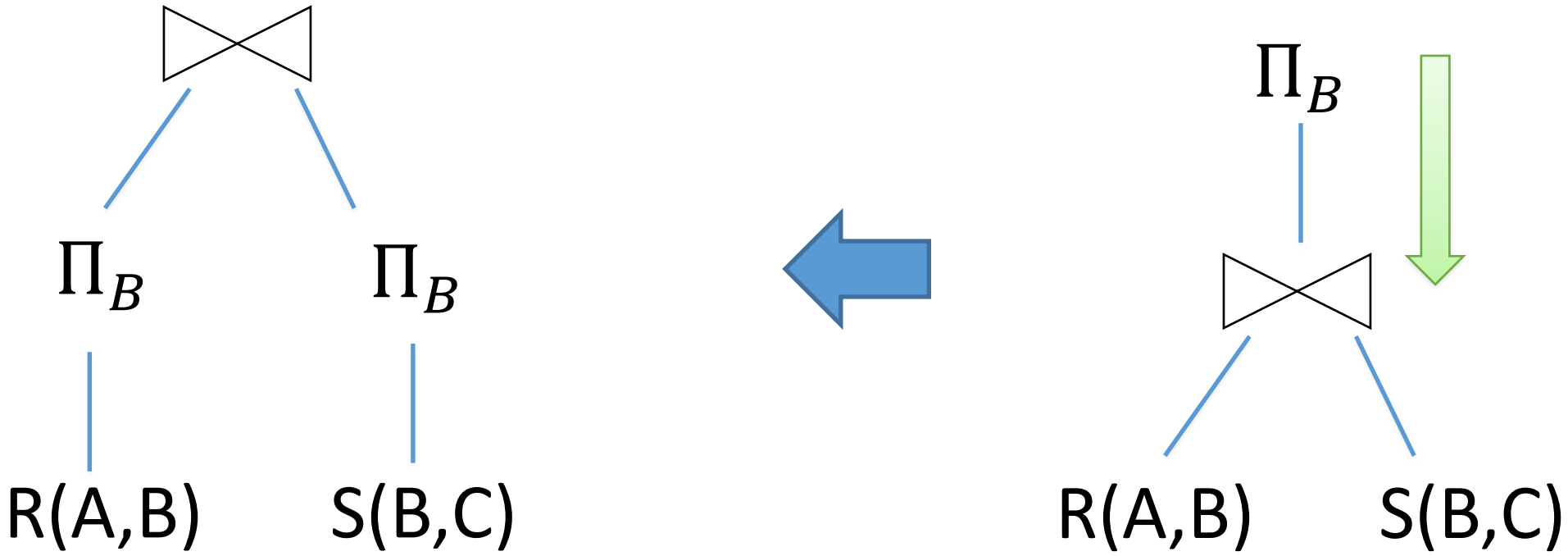
A simple plan

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.