Updated 2/8/2022

Topic 1: Data models and query languages
Unit 3: Relational Algebra (continued)
Lecture 07

Wolfgang Gatterbauer

CS7240 Principles of scalable data management (sp22)

https://northeastern-datalab.github.io/cs7240/sp22/

2/8/2022

Pre-class conversations

- Last class recapitulation
- Please keep on pointing out any errors on the slides
- Project discussions
- Where we are

- today:
  - Algebra (cont), Codd's theorem

Topic 1: Data Models and Query Languages

- **Lecture 1 (Tue 1/18):** Course introduction / SQL / PostgreSQL setup / SQL Activities
- **Lecture 2 (Fri 1/21):** SQL
- **Lecture 3 (Tue 1/25):** SQL
- **Lecture 4 (Fri 1/28):** SQL, Logic & Relational Calculus
- **Lecture 5 (Tue 2/1):** Logic & Relational Calculus
- **Lecture 6 (Fri 2/4):** Relational Algebra & Codd's Theorem
- **Lecture 7 (Tue 2/8):** Relational Algebra & Codd's Theorem / Datalog & Recursion
- **Lecture 8 (Fri 2/11):** Datalog & Recursion
- **Lecture 9 (Tue 2/15):** Alternative Data Models

Pointers to relevant concepts & supplementary material:

- **Unit 1. SQL**: [SAMS'12], [CS 3200], [Cow'03] Ch3 & Ch5, [Complete'08] Ch6, [Silberschatz+'20] Ch3.8
- **Unit 2. Logic & Relational Calculus**: First-Order Logic (FOL), relational calculus (RC): [Barland+'08] 4.1.2 & 4.2.1 & 4.4, [Genesereth+] Ch6, [Halpern+'01], [Cow'03] Ch4.3 & 4.4, [Elmasri, Navathe'15] Ch8.6 & Ch8.7, [Silberschatz+'20] Ch27.1 & Ch27.2, [Alice'95] Ch3.1-3.3 & Ch4.2 & Ch4.4 & Ch5.3-5.4, [Barker-Plummer+'11] Ch11
- **Unit 3. Relational Algebra & Codd's Theorem**: Relational Algebra (RA), Codd's theorem: [Cow'03] Ch4.2, [Complete'08] Ch2.4 & Ch5.1-5.2, [Elmasri, Navathe'15] Ch8, [Silberschatz+'20] Ch2.6, [Alice'95] Ch4.4 & Ch5.4
- **Unit 4. Datalog & Recursion**: Datalog, recursion, Stratified Datalog with negation, Datalog evaluation strategies, Stable Model semantics, Answer Set Programming (ASP): [Complete'08] Ch5.3, [Cow'03] Ch 24, [Koutris'19] L9 & L10, [G., Suciu'10]
- **Unit 5. Alternative Data Models**: NoSQL: [Hellerstein, Stonebraker'05], [Sadalage, Fowler'12], [Harrison'16]

Commuting functions: a digression

Side-topic

- Do functions commute with taking the expectation?
  - $\mathbb{E}[f(x)] = f(\mathbb{E}[x])$?

Commuting functions: a digression

Side-topic

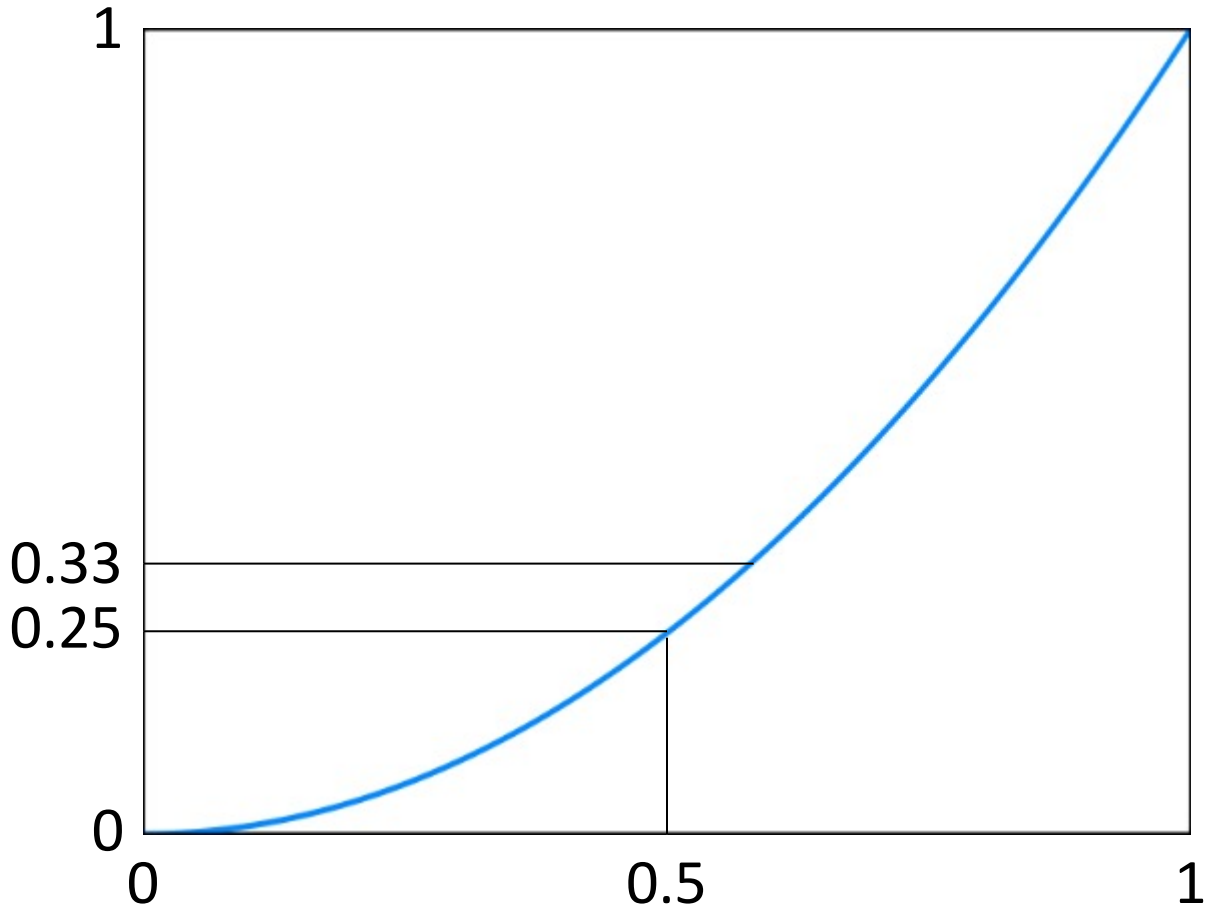
- Do functions commute with taking the expectation?
  - $\mathbb{E}[f(x)] = f(\mathbb{E}[x])$?
- Only for linear functions
  - Thus $f(x)=ax + b$
  - $\mathbb{E}[ax+b] = a\ \mathbb{E}[x] + b$
- Jensen's inequality for convex f

?

Commuting functions: a digression

Side-topic

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



Ratio of averages != average of ratios

Side-topic

	Variant 1	Variant 2	Ratio $\frac{\text{Variant 1}}{\text{Variant 2}}$
North	20	10	20/10 = 2
South	10	20	10/20 = 0.5

AVG = 1.25 + 25%

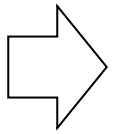
Variant 1 is on average 25% better across North and South

RA Operators are Compositional, in general

Student(sid,sname,gpa)

How do we represent this query in RA?

```
SELECT DISTINCT sname, gpa
FROM Student
WHERE gpa > 3.5
```



?

RA Operators are Compositional, in general

Student(sid,sname,gpa)

```
SELECT DISTINCT sname, gpa
FROM Student
WHERE gpa > 3.5
```

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

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

Which of those two variants is correct?

RA Operators are Compositional, in general

Student(sid,sname,gpa)

```
SELECT DISTINCT sname, gpa
FROM Student
WHERE gpa > 3.5
```

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

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

Both are correct: logically equivalent ☺

Relational Algebra (RA) operators

- Five basic operators:
  1. Selection: σ ("sigma")
  2. Projection: Π
  3. Cartesian Product: $\times$
  4. Union: $\cup$
  5. Difference: $-$
- Auxiliary (or special) operator
  6. Renaming: ρ ("rho")
- Derived (or implied) operators
  7. Joins $\bowtie$ (natural, theta join, equi-join, semi-join)
  8. Intersection / complement
  9. Division

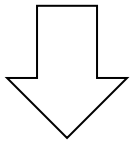
3. Cross-Product (×)

- Each tuple in R with each tuple in S
- Notation: R × S
- Example:
  - Students × Advisors
- Rare in practice; mainly used to express joins

Student(sid,sname,gpa)
People(ssn,pname,address)

SQL:

```
SELECT *
FROM People, Student
```



RA:

?

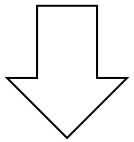
3. Cross-Product (×)

- Each tuple in R with each tuple in S
- Notation: R × S
- Example:
  - Students × Advisors
- Rare in practice; mainly used to express joins

Student(sid,sname,gpa)
People(ssn,pname,address)

SQL:

```
SELECT *
FROM People, Student
```



RA:

$$\text{People} \times \text{Student}$$

3. Cross join example

People

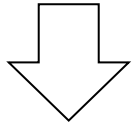
ssn	pname	address
1234545	John	216 Rosse
5423341	Bob	217 Rosse

×

Student

sid	sname	gpa
001	John	3.4
002	Bob	1.3

People × Student



?

3. Cross join example

People

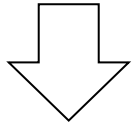
ssn	pname	address
1234545	John	216 Rosse
5423341	Bob	217 Rosse

×

Student

sid	sname	gpa
001	John	3.4
002	Bob	1.3

People × Student



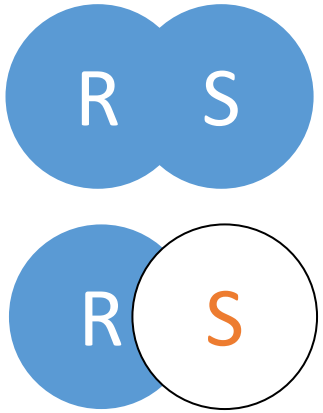
ssn	pname	address	sid	sname	gpa
1234545	John	216 Rosse	001	John	3.4
~~5423341~~	~~Bob~~	~~217 Rosse~~	~~001~~	~~John~~	~~3.4~~
~~1234545~~	~~John~~	~~216 Rosse~~	~~002~~	~~Bob~~	~~1.3~~
5423341	Bob	216 Rosse	002	Bob	1.3

Relational Algebra (RA) operators

- Five basic operators:
  1. Selection: σ ("sigma")
  2. Projection: Π
  3. Cartesian Product: $\times$
  4. Union: $\cup$
  5. Difference: $-$
- Auxiliary (or special) operator
  6. Renaming: ρ ("rho")
- Derived (or implied) operators
  7. Joins $\bowtie$ (natural, theta join, equi-join, semi-join)
  8. Intersection / complement
  9. Division

4. Union (U) and 5. Difference (–)

R ∪ S
R – S



- Examples:
  - Students ∪ Faculty
  - AllNEUEmployees – RetiredFaculty

Student (<u>neuid</u>, fname, lname)
Faculty (<u>neuid</u>, fname, lname, college)

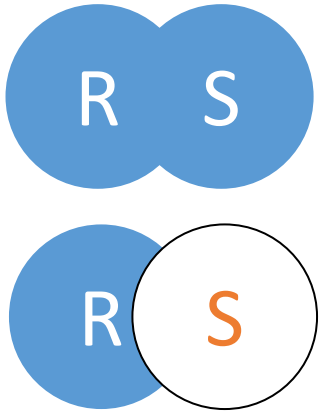
What about the union of Student and Faculty?

Difference also sometimes written with "\", thus as R\S

4. Union (U) and 5. Difference (–)

Actor (aid, fname, lname)
Play (aid, mid, role)

R ∪ S
R – S



Other example: find actor ids who don't play in any movie:

?

- Examples:
  - Students ∪ Faculty
  - AllNEUEmployees – RetiredFaculty

Student (neuid, fname, lname)
$\pi_{\text{-college}}$(Faculty (neuid, fname, lname, college))

What about the union of Student and Faculty?

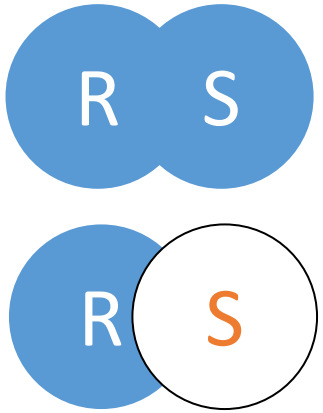
No! Only makes sense if R and S are "compatible", thus have the same schema!

Difference also sometimes written with "\", thus as R\S

4. Union (∪) and 5. Difference (–)

Actor (aid, fname, lname)
Play (aid, mid, role)

R ∪ S
R – S



Other example: find actor ids who don't play in any movie:

$$\pi_{aid}(\text{Actor}) - \pi_{aid}(\text{Play})$$

- Examples:
  - Students ∪ Faculty
  - AllNEUEmployees – RetiredFaculty

$\pi_{\text{-college}}$(Student (neuid, fname, lname)
Faculty (neuid, fname, lname, college))

What about the union of Student and Faculty?

No! Only makes sense if R and S are "compatible", thus have the same schema!

Difference also sometimes written with "\", thus as R\S

Relational Algebra (RA) operators

- Five basic operators:
  1. Selection: σ ("sigma")
  2. Projection: Π
  3. Cartesian Product: $\times$
  4. Union: $\cup$
  5. Difference: $-$
- Auxiliary (or special) operator
  6. Renaming: ρ ("rho")
- Derived (or implied) operators
  7. Joins $\bowtie$ (natural, theta join, equi-join, semi-join)
  8. Intersection / complement
  9. Division

6. Renaming (ρ rho)

- Does not change the instance, only the schema (table or attribute names)
- Only needed in named perspective, thus a 'special' operator (neither basic nor derived)
- Several existing conventions:

$\rho_S(R)$ S new table name

$\rho_{S(B_1,\ldots,B_n)}(R)$ if positions can be used

$\rho_{S(A_1 \to B_1,\ldots,A_n \to B_n)}(R)$

$\rho_{A_1 \to B_1,\ldots,A_n \to B_n}(R)$ if attribute names, not order matters

$\rho_{B_1,\ldots,B_n}(R)$

Student(<u>sid</u>,sname,gpa)

SQL:

```
SELECT
    sid      AS studId,
    sname AS name,
    gpa      AS gradePtAvg
FROM Student
```

RA:

?

Alternative to "→ " is the substitution symbol "/ ", e.g. B_1/A_1 same as $A_1 \to B_1$

6. Renaming (ρ rho)

- Does not change the instance, only the schema (table or attribute names)
- Only needed in named perspective, thus a 'special' operator (neither basic nor derived)
- Several existing conventions:

$\rho_S(R)$ S new table name

$\rho_{S(B_1,\ldots,B_n)}(R)$ if positions can be used

$\rho_{S(A_1 \to B_1,\ldots,A_n \to B_n)}(R)$ if attribute names, not order matters

$\rho_{A_1 \to B_1,\ldots,A_n \to B_n}(R)$

$\rho_{B_1,\ldots,B_n}(R)$

Student(<u>sid</u>,sname,gpa)

SQL:

```
SELECT
    sid      AS studId,
    sname AS name,
    gpa      AS gradePtAvg
FROM Student
```

RA:

$\rho_{\text{studId,name,gradePtAvg}}(\text{Student})$

Alternative to "→ " is the substitution symbol "/ ", e.g. B_1/A_1 same as $A_1 \to B_1$

6. Why we need renaming

R

A	B
1	2
3	4

S

B	C	D
2	5	6
4	7	8
9	10	11

$R \times S$

?

6. Why we need renaming

R

A	B
1	2
3	4

S

B	C	D
2	5	6
4	7	8
9	10	11

$R \times S$

A	R.B	S.B	C	D
1	2	2	5	6
1	2	4	7	8
1	2	9	10	11
3	4	2	5	6
3	4	4	7	8
3	4	9	10	11

But what if we had $R \times R$?

6. Why we need renaming

R

A	B → E
1	2
3	4

S

B	C	D
2	5	6
4	7	8
9	10	11

$R \times S$

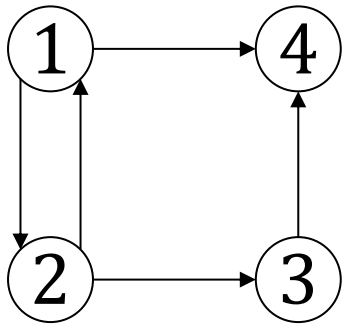
A	R.B	S.B	C	D
1	2	2	5	6
1	2	4	7	8
1	2	9	10	11
3	4	2	5	6
3	4	4	7	8
3	4	9	10	11

$\rho_{B \to E}(R) \times S$

A	E	B	C	D
1	2	2	5	6
1	2	4	7	8
1	2	9	10	11
3	4	2	5	6
3	4	4	7	8
3	4	9	10	11

But what if we had $R \times R$?

6. Named vs Unnamed perspective on a Graph



Q: Nodes that have a grand-child $\{1,2\}$

In RC:

?

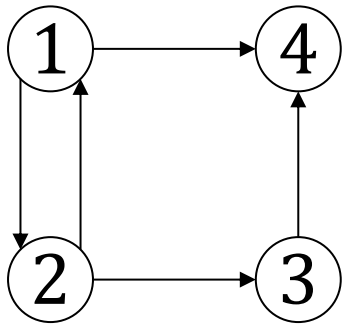
A:

S	T
1	2
2	1
2	3
1	4
3	4

In RA:

?

6. Named vs Unnamed perspective on a Graph



Q: Nodes that have a grand-child {1,2}

In RC:

$$\{ x \mid \exists y,z.[A(x,y) \wedge A(y,z)] \}$$

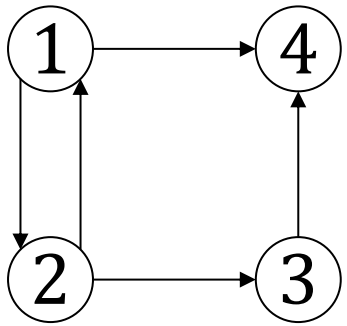
$$\{ x \mid \exists y,z,u,w.[A(y,z) \wedge A(u,w) \wedge z=u \wedge y=x] \}$$

A:	S	T
	1	2
	2	1
	2	3
	1	4
	3	4

In RA:

?

6. Named vs Unnamed perspective on a Graph



Q: Nodes that have a grand-child {1,2}

In RC:

$$\{ x \mid \exists y,z.[A(x,y) \wedge A(y,z)] \}$$ "unnamed"

$$\{ x \mid \exists y,z,u,w.[A(y,z) \wedge A(u,w) \wedge z=u \wedge y=x] \}$$

?

"named"

In RA:

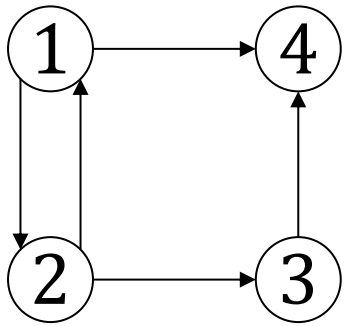
$$\pi_S(\sigma_{T=S2}(A \times \rho_{S \to S2}(A)))$$ named perspective

?

unnamed perspective

A:	S	T
	1	2
	2	1
	2	3
	1	4
	3	4

6. Named vs Unnamed perspective on a Graph



A:

S	T
1	2
2	1
2	3
1	4
3	4

Q: Nodes that have a grand-child $\{1,2\}$

In RC:

DRC $\{ x \mid \exists y,z.[A(x,y) \wedge A(y,z)]\}$ "unnamed"

$\{ x \mid \exists y,z,u,w.[A(y,z) \wedge A(u,w) \wedge z=u \wedge y=x]\}$

TRC $\{ q(S) \mid \exists a1, a2 \in A[a1.T=a2.S \wedge a1.S=q.S]\}$ "named"

In RA:

$\pi_S(\sigma_{T=S2}(A \times \rho_{S \to S2}(A)))$ named perspective

$\pi_{\$1}(\sigma_{\$2=\$3}(A \times A))$ unnamed perspective

\$2 was used in Ullman's old textbook. Often just written as "$\pi_1(\sigma_{2=3}(A \times A))$". A more recent database textbook uses "$2 \doteq 3$" for "\$2=\$3" which gets confusing for "\$2=3"...

Another example

Employee(<u>id</u>, name, salary)

Q: *Find the ID and name of those employees who earn more than the employee whose ID is 123?*

?

Another example

Employee(<u>id</u>, name, salary)

Q: *Find the ID and name of those employees who earn more than the employee whose ID is 123?*

$$\pi_{\text{e.id,e.name}}\left(\sigma_{\text{e.salary}>\text{o.salary}}(\rho_{\text{e}}(\text{employee})\times\sigma_{\text{id}=123}(\rho_{\text{o}}(\text{employee})))\right)$$

Relational Algebra (RA) operators

- Five basic operators:
  1. Selection: σ ("sigma")
  2. Projection: Π
  3. Cartesian Product: $\times$
  4. Union: $\cup$
  5. Difference: $-$
- Auxiliary (or special) operator
  6. Renaming: ρ ("rho")
- Derived (or implied) operators
  7. Joins $\bowtie$ (natural, theta join, equi-join, semi-join)
  8. Intersection / complement
  9. Division

Derived relational operators:
- can be expressed in basic RA; thus not needed

But enhancing the basic operator set with derived operators is a good idea:
- Queries become easier to write/understand/maintain
- Easier for DBMS to apply specialized optimizations (recall the conceptual evaluation strategy)

most important

we discuss later in class in detail

7a. Natural Join (⋈)

Product(pname, price, category, cid)
Company(cid, cname, stockprice, country)

- Notation: R ⋈ S
- Joins R and S on equality of all shared attributes
  - Only makes sense in named perspective!
  - If R has attribute set A, and S has attribute set B, and they share attributes A∩B = C, can also be written as $R \bowtie_C S$
- Natural join in basic RA:
  - Meaning: $R \bowtie S = \Pi_{A \cup B}(\sigma_{R.C=S.C}(R \times S))$
  - Meaning: $R \bowtie S = \Pi_{A \cup B}(\sigma_{C=D}(\rho_{C \to D}(R) \times S))$
    - 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

SQL

```
SELECT pname, price, category,
P.cid, cname, stockprice, country
FROM Product P, Company C
WHERE P.cid= C.cid
```

SQL (alternative syntax)

?

7a. Natural Join (⋈)

Product(pname, price, category, cid)
Company(cid, cname, stockprice, country)

- Notation: R ⋈ S
- Joins R and S on equality of all shared attributes
  - Only makes sense in named perspective!
  - If R has attribute set A, and S has attribute set B, and they share attributes A∩B = C, can also be written as R $\bowtie_C$ S

- Natural join in basic RA:
  - Meaning: R ⋈ S = $\Pi_{A \cup B}(\sigma_{R.C=S.C}(R \times S))$
  - Meaning: R ⋈ S = $\Pi_{A \cup B}(\sigma_{C=D}(\rho_{C \to D}(R) \times S))$
    - 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

SQL

```
SELECT pname, price, category,
P.cid, cname, stockprice, country
FROM Product P, Company C
WHERE P.cid= C.cid
```

SQL (alternative syntax)

```
SELECT *
FROM Product
NATURAL JOIN Company
```

RA:

7a. Natural Join (⋈)

Product(pname, price, category, cid)
Company(cid, cname, stockprice, country)

- Notation: R ⋈ S
- Joins R and S on equality of all shared attributes
  - Only makes sense in named perspective!
  - If R has attribute set A, and S has attribute set B, and they share attributes A∩B = C, can also be written as R $\bowtie_C$ S

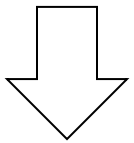
- Natural join in basic RA:
  - Meaning: R ⋈ S = $\Pi_{A \cup B}(\sigma_{R.C=S.C}(R \times S))$
  - Meaning: R ⋈ S = $\Pi_{A \cup B}(\sigma_{C=D}(\rho_{C \to D}(R) \times S))$
    - The rename $\rho_{C \to D}$ renames the shared attributes in one of the relations
    - The selection $\sigma_{C=D}$ checks equality of the shared attributes
    - The projection $\Pi_{A \cup B}$ eliminates the duplicate common attributes

SQL

```
SELECT pname, price, category,
P.cid, cname, stockprice, country
FROM Product P, Company C
WHERE P.cid= C.cid
```

SQL (alternative syntax)

```
SELECT *
FROM Product
NATURAL JOIN Company
```



RA:

$$\text{Product} \bowtie \text{Company}$$

7a. Natural Join (⋈): an alternative perspective

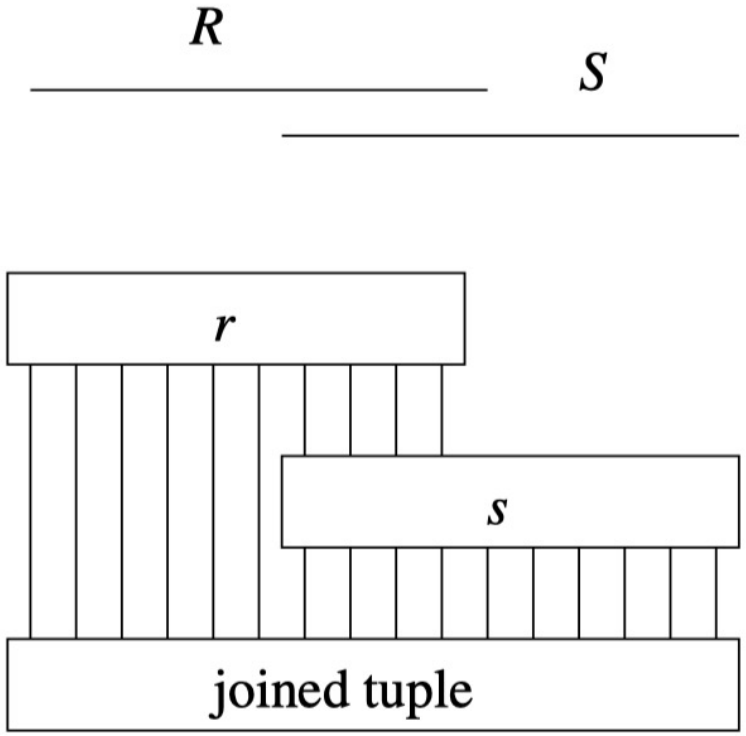


Figure 15: Joining tuples

More formally the semantics of the natural join are defined as follows:

$$R \bowtie S = \{r \cup s \mid r \in R \ \wedge \ s \in S \ \wedge \ Fun(r \cup s)\} \quad (1)$$

where *Fun(t)* is a predicate that is true for a relation *t* (in the mathematical sense) iff *t* is a function. It is usually required that *R* and *S* must have at least one common attribute, but if this constraint is omitted, and *R* and *S* have no common attributes, then the natural join becomes exactly the Cartesian product.

Sources: Garcia-Molina, Ullman, Widom. Database Systems -- The Complete Book (2nd ed, international ed), 2014. http://infolab.stanford.edu/~ullman/dscb.html , https://en.wikipedia.org/wiki/Relational_algebra#Natural_join
Wolfgang Gatterbauer. Principles of scalable data management: https://northeastern-datalab.github.io/cs7240/

7a. Natural Join (⋈): An example

R

A	B
1	2
3	4

S

B	C	D
2	5	6
4	7	8
9	10	11

$$\rho_{B \to E}(\mathrm{R}) \times \mathrm{S}$$

?

7a. Natural Join (⋈): An example

R

A	B
1	2
3	4

S

B	C	D
2	5	6
4	7	8
9	10	11

$R \bowtie S$

?

$\rho_{B \to E}(R) \times S$

A	E	B	C	D
1	2	2	5	6
1	2	4	7	8
1	2	9	10	11
3	4	2	5	6
3	4	4	7	8
3	4	9	10	11

7a. Natural Join (⋈): An example

R

A	B
1	2
3	4

S

B	C	D
2	5	6
4	7	8
9	10	11

$R \bowtie S$

A	B	C	D
1	2	5	6
3	4	7	8

$R \bowtie S =$ in basic RA

?

$\rho_{B \to E}(R) \times S$

A	E	B	C	D
1	2	2	5	6
1	2	4	7	8
1	2	9	10	11
3	4	2	5	6
3	4	4	7	8
3	4	9	10	11

7a. Natural Join (⋈): An example

R

A	B
1	2
3	4

S

B	C	D
2	5	6
4	7	8
9	10	11

R ⋈ S

A	B	C	D
1	2	5	6
3	4	7	8

R ⋈ S =

$\Pi_{ABCD}(\sigma_{R.B=S.B}(R \times S))$ =

$\Pi_{AR.BCD}(\sigma_{R.B=S.B}(R \times S))$ =

$\Pi_{ABCD}(\sigma_{B=E}(\rho_{B\rightarrow E}(R) \times S))$

$\rho_{B\rightarrow E}(R) \times S$

A	E	B	C	D
1	2	2	5	6
1	2	4	7	8
1	2	9	10	11
3	4	2	5	6
3	4	4	7	8
3	4	9	10	11

7a. Natural Join (⋈): practice

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

?

7a. Natural Join (⋈): practice

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

  Answer(A, B, C, D,E)

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

  ?

7a. Natural Join (⋈): practice

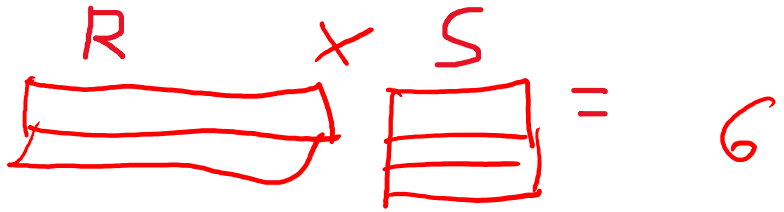
- Given schemas R(A, B, C, D), S(A, C, E), what is the schema of R ⋈ S ?

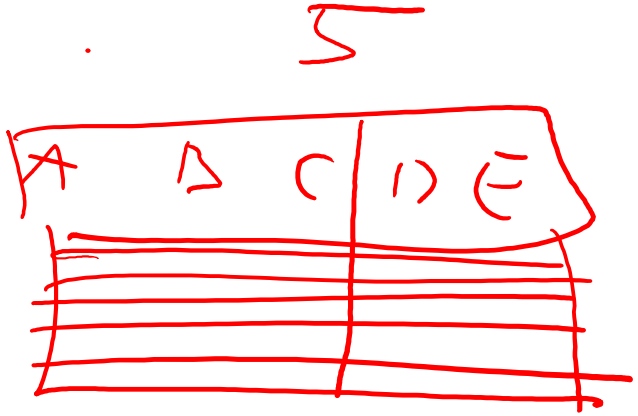
  Answer(A, B, C, D,E)

no condition in the selection that could be violated:

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

  R × S





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

?

7a. Natural Join (⋈): practice

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

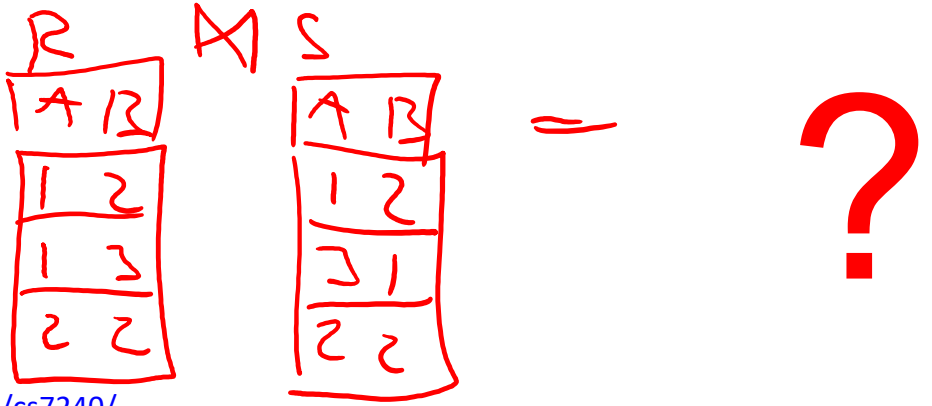
  Answer(A, B, C, D,E)

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

  R × S

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

  R ∩ S



7a. Natural Join (⋈): practice

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

  Answer(A, B, C, D,E)

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

  R × S

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

  R ∩ S

7b. Theta Join ($\bowtie_\theta$)

- A join that involves a predicate

$$R_1 \bowtie_\theta R_2 = \sigma_\theta(R_1 \times R_2)$$

- θ ("theta") can be any condition
- No projection: #attributes in output = sum #attributes in input

Note that natural join is a theta join + a selection

- Example: band-joins for approx. matchings across tables

AnonPatient (age, zip, disease)
Voters (name, age, zip)

Assume relatively fresh data (within 1 year)

?

7b. Theta Join ($\bowtie_\theta$)

Student(sid,name,gpa)
People(ssn,name,address)

- A join that involves a predicate

$$R_1 \bowtie_\theta R_2 = \sigma_\theta(R_1 \times R_2)$$

- θ ("theta") can be any condition
- No projection: #attributes in output = sum #attributes in input

Note that natural join is a theta join + a selection

- Example: band-joins for approx. matchings across tables

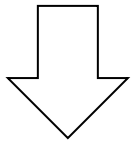
AnonPatient (age, zip, disease)
Voters (name, age, zip)

Assume relatively fresh data (within 1 year)

$$A \bowtie_{P.zip=V.zip \wedge P.age>=V.age\ -1 \wedge P.age<=V.age\ +1} V$$

SQL:

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



RA:

?

7b. Theta Join ($\bowtie_\theta$)

Student(sid,name,gpa)
People(ssn,name,address)

- A join that involves a predicate

$$R_1 \bowtie_\theta R_2 = \sigma_\theta(R_1 \times R_2)$$

- θ ("theta") can be any condition
- No projection: #attributes in output = sum #attributes in input

Note that natural join is a theta join + a selection

- Example: band-joins for approx. matchings across tables

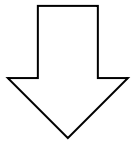
AnonPatient (age, zip, disease)
Voters (name, age, zip)

Assume relatively fresh data (within 1 year)

$$A \bowtie_{P.zip=V.zip \wedge P.age>=V.age-1 \wedge P.age<=V.age+1} V$$

SQL:

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



RA:

$$Students \bowtie_\theta People$$

7c. Equi-join ($\bowtie_{A=B}$)

Student(sid,sname,gpa)
People(ssn,pname,address)

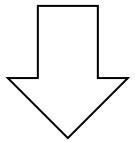
- A theta join where q is an equality

$$R_1 \bowtie_{A=B} R_2 = \sigma_{A=B}(R_1 \times R_2)$$

- Example over Gizmo DB:
  - Product $\bowtie_{manufacturer=cname}$ Company
- Most common join in practice!

SQL:

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



RA:

?

7c. Equi-join ($\bowtie_{A=B}$)

Student(sid,sname,gpa)
People(ssn,pname,address)

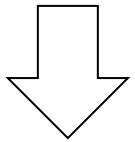
- A theta join where q is an equality

$$R_1 \bowtie_{A=B} R_2 = \sigma_{A=B}(R_1 \times R_2)$$

- Example over Gizmo DB:
  - Product $\bowtie_{\text{manufacturer=cname}}$ Company
- Most common join in practice!

SQL:

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



RA:

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

What is the connection with a natural join?

Join Summary

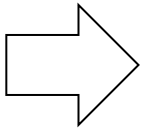
- **Theta-join**: $R \bowtie_\theta S = \sigma_\theta (R \times S)$
  - Join of R and S with a join condition θ
  - Cross-product followed by selection θ
  - No projection
- **Equijoin**: $R \bowtie_\theta S = \sigma_\theta (R \times S)$
  - Join condition θ consists only of equalities
  - No projection
- **Natural join**: $R \bowtie S = \pi_A (\sigma_\theta (R \times S))$
  - Equality on **all** fields with same name in R and in S
  - Projection π_A drops all redundant attributes

Example: Converting SFW Query -> RA

Student(sid,name,gpa)
People(ssn,name,address)

```
SELECT DISTINCT gpa, address
FROM Student S, People P
WHERE S.name = P.name
AND gpa > 3.5
```

How do we represent this query in RA?



?

Example: Converting SFW Query -> RA

Student(sid,name,gpa)
People(ssn,name,address)

```
SELECT DISTINCT gpa, address
FROM Student S, People P
WHERE S.name = P.name
AND gpa > 3.5
```

How do we represent this query in RA?

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

$$\Pi_{\text{gpa,address}}(\sigma_{\text{gpa}>3.5 \wedge \text{S.name}=\text{P.name}}(S \times P))$$

$$\Pi_{\text{gpa,address}}(\sigma_{\text{gpa}>3.5 \wedge \text{name}=\text{name2}}(S \times \rho_{\text{name}\rightarrow\text{name2}} P))$$

Some Examples

Supplier(sno,sname,scity,sstate)
Part(pno,pname,psize,pcolor)
Supply(sno,pno,qty,price)

Name of supplier of parts with size greater than 10

?

Name of supplier of red parts or parts with size greater than 10

?

Some Examples

Supplier(sno,sname,scity,sstate)
Part(pno,pname,psize,pcolor)
Supply(sno,pno,qty,price)

Name of supplier of parts with size greater than 10

$\pi_{sname}(\sigma_{psize>10}(Supplier \bowtie Supply \bowtie Part))$

$\pi_{sname}(Supplier \bowtie Supply \bowtie (\sigma_{psize>10}(Part)))$

Name of supplier of red parts or parts with size greater than 10

?

Some Examples

Supplier(sno,sname,scity,sstate)
Part(pno,pname,psize,pcolor)
Supply(sno,pno,qty,price)

Name of supplier of parts with size greater than 10

$\pi_{sname}(\sigma_{psize>10}(Supplier \bowtie Supply \bowtie Part))$

$\pi_{sname}(Supplier \bowtie Supply \bowtie (\sigma_{psize>10}(Part))$ → Representation of RA as tree?

Name of supplier of red parts or parts with size greater than 10

$\pi_{sname}(Supplier \bowtie Supply \bowtie (\sigma_{psize>10}(Part) \cup \sigma_{pcolor='red'}(Part)))$

$\pi_{sname}(Supplier \bowtie Supply \bowtie (\sigma_{psize>10 \vee pcolor='red'}(Part)))$

Some Examples

Supplier(<u>sno</u>,sname,scity,sstate)
Part(<u>pno</u>,pname,psize,pcolor)
Supply(<u>sno,pno</u>,qty,price)

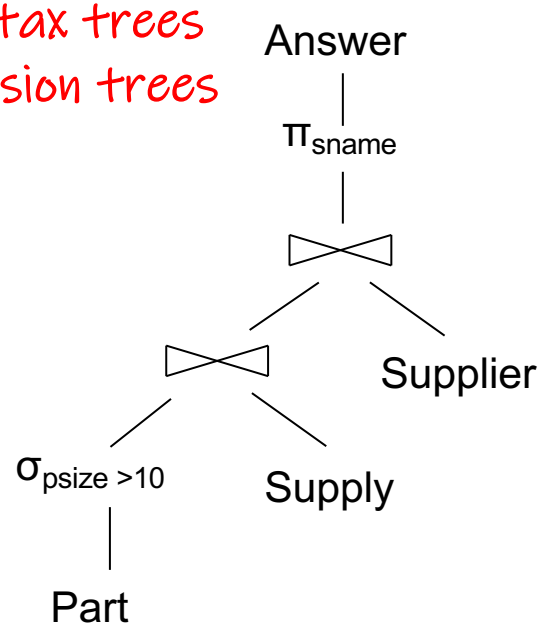
Usually unary or binary. Think of:
- abstract syntax trees
- binary expression trees
- parse trees

Name of supplier of parts with size greater than 10

$\pi_{sname}(\sigma_{psize>10}(Supplier \bowtie (Supply \bowtie Part)))$

$\pi_{sname}(Supplier \bowtie Supply \bowtie (\sigma_{psize>10}(Part)))$

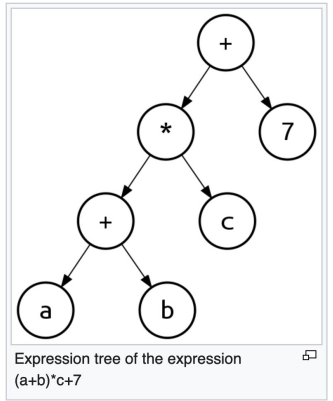
Representation of RA as tree?



Name of supplier of red parts or parts with size greater than 10

$\pi_{sname}(Supplier \bowtie Supply \bowtie (\sigma_{psize>10}(Part) \cup \sigma_{pcolor='red'}(Part)))$

$\pi_{sname}(Supplier \bowtie Supply \bowtie (\sigma_{psize>10 \vee pcolor='red'}(Part)))$



Expression tree of the expression (a+b)*c+7

https://en.wikipedia.org/wiki/Binary_expression_tree

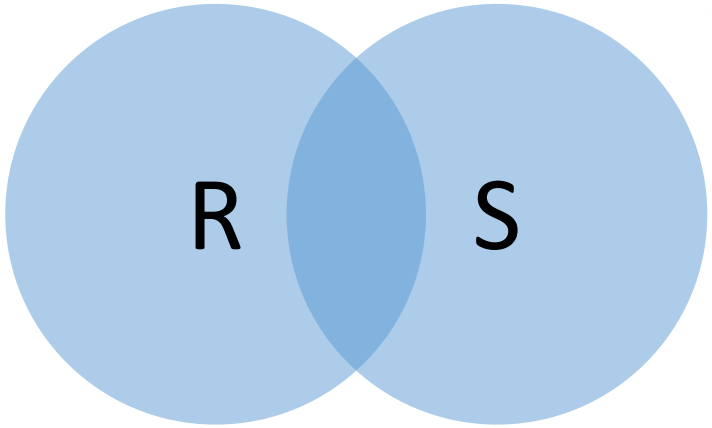
Relational Algebra (RA) operators

- Five basic operators:
  1. Selection: σ ("sigma")
  2. Projection: Π
  3. Cartesian Product: $\times$
  4. Union: $\cup$
  5. Difference: $-$
- Auxiliary (or special) operator
  6. Renaming: ρ ("rho")
- Derived (or implied) operators
  7. Joins $\bowtie$ (natural, theta join, equi-join, semi-join)
  8. Intersection / complement
  9. Division

8. What about Intersection ∩?

- As derived operator using union and minus

  ?



8. What about Intersection ∩?

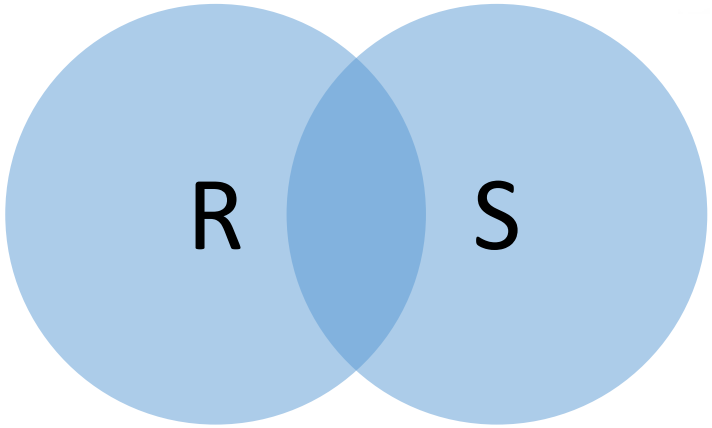
- As derived operator using union and minus

  $R \cap S = ((R \cup S) - (R - S)) - (S - R)$

  $R \cap S = (R \cup S) - ((R - S) \cup (S - R))$

- Derived operator using minus only!

  ?



8. What about Intersection ∩?

- As derived operator using union and minus

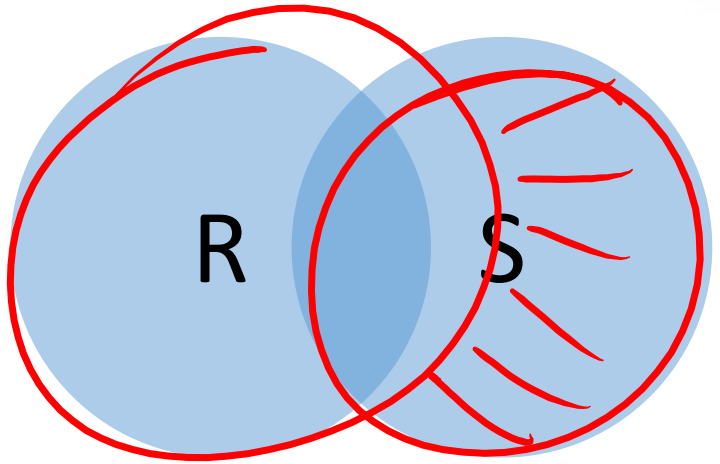
$$R \cap S = ((R \cup S) - (R - S)) - (S - R)$$
$$R \cap S = (R \cup S) - ((R - S) \cup (S - R))$$

- Derived operator using minus only!

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

- Derived using join

?

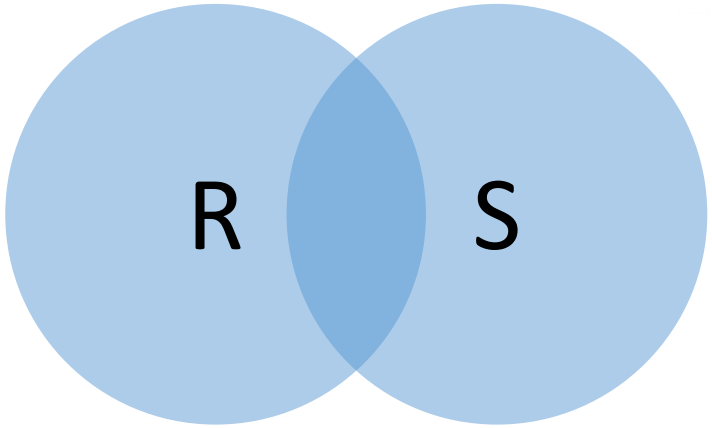


S - (S - R)

8. What about Intersection ∩?

- As derived operator using union and minus

$$R \cap S = ((R \cup S) - (R - S)) - (S - R)$$
$$R \cap S = (R \cup S) - ((R - S) \cup (S - R))$$



- Derived operator using minus only!

$$R \cap S = \quad S \quad - (S - R)$$

- Derived using join

$$R \cap S = R \bowtie S$$

Schemas need to be compatible:
not
R(A,B,C)
S(A,B)

Relational Algebra (RA) operators

- Five basic operators:
  1. Selection: σ ("sigma")
  2. Projection: Π
  3. Cartesian Product: $\times$
  4. Union: $\cup$
  5. Difference: $-$
- Auxiliary (or special) operator
  6. Renaming: ρ ("rho")
- Derived (or implied) operators
  7. Joins $\bowtie$ (natural, theta join, equi-join, semi-join)
  8. Intersection / complement
  9. Division

9. Division (R ÷ S)

- Consider two relations R(X,Y) and S(Y)
- Then R ÷ S is ...

X, Y are sets of attributes

What could be a meaningful definition of division ?

Compare to Integer division: 7/2=3

9. Division (R ÷ S)

- Consider two relations R(X,Y) and S(Y) X, Y are sets of attributes
- Then R ÷ S is ...
  - ... the largest relation T(X) s.t. S × T ⊆ R, or (safety: T ⊆ π_XR)
  - ... the relation T(X) that contains the X's that occur with all Y's in S

Dividend

R

X	Y
Alice	1
Alice	2
Bob	1
Bob	2
Bob	3

Divisor

S

Y
1
2
3

T

?

9. Division (R ÷ S)

- Consider two relations R(X,Y) and S(Y) X, Y are sets of attributes
- Then R ÷ S is ...
  - ... the largest relation T(X) s.t. S × T ⊆ R, or (safety: T ⊆ π_XR)
  - ... the relation T(X) that contains the X's that occur with all Y's in S

Dividend

R

X	Y
Alice	1
Alice	2
Bob	1
Bob	2
Bob	3

Divisor

S

Y
1
2
3

T

X
Bob

9. Division: More formal Definition

- Legal input: (R,S) such that R has all the attributes of S and more
- R÷S is the relation T with:
  - The header of R, with all attributes of S removed
  - Tuple set $\{t[\mathbf{X}] \mid t[\mathbf{X},\mathbf{Y}] \in R \text{ for all } s[\mathbf{Y}] \in S\}$

Notice the different notation for projection here (t[X] instead of t.X).
Also minor abuse of notation, since the attributes in X need not necessarily come before those of Y

Questions

Studies

sid	student	course
1	Alice	AI
1	Alice	DB
2	Bob	DB
2	Bob	ML
3	Charly	AI
3	Charly	DB
3	Charly	ML

÷

Course

course
ML

= ?

÷

course
AI
DB
ML

= ?

Questions

recall set semantics for RA

Studies

sid	student	course
1	Alice	AI
1	Alice	DB
2	Bob	DB
2	Bob	ML
3	Charly	AI
3	Charly	DB
3	Charly	ML

÷

Course

course
ML

=

sid	student
2	Bob
3	Charly

÷

course
AI
DB
ML

=

sid	student
3	Charly

Assume R,S have disjoint attribute sets (possibly by renaming)

(RxS)÷S = ?

(RxS)÷R = ?

Questions

recall set semantics for RA

Studies

sid	student	course
1	Alice	AI
1	Alice	DB
2	Bob	DB
2	Bob	ML
3	Charly	AI
3	Charly	DB
3	Charly	ML

÷

Course

course
ML

=

sid	student
2	Bob
3	Charly

÷

course
AI
DB
ML

=

sid	student
3	Charly

Assume R,S have disjoint attribute sets (possibly by renaming)

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

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

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?

?

Questions

recall set semantics for RA

Studies

sid	student	course
1	Alice	AI
1	Alice	DB
2	Bob	DB
2	Bob	ML
3	Charly	AI
3	Charly	DB
3	Charly	ML

÷

Course

course
ML

=

sid	student
2	Bob
3	Charly

÷

course
AI
DB
ML

=

sid	student
3	Charly

Assume R,S have disjoint attribute sets (possibly by renaming)

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

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

Q: If R has 1000 tuples and S has 100 tuples, how many tuples can be in R÷S? 0-10

Q: If R has 1000 tuples and S has 1001 tuples, how many tuples can be in R÷S? 0

Questions

Studies

sid	student	course
1	Alice	AI
1	Alice	DB
2	Bob	DB
2	Bob	ML
3	Charly	AI
3	Charly	DB
3	Charly	ML

CourseType

course	type
AI	elective
DB	core
ML	core

Who took all core courses in RA with relational division?

?

Questions

Studies

sid	student	course
1	Alice	AI
1	Alice	DB
2	Bob	DB
2	Bob	ML
3	Charly	AI
3	Charly	DB
3	Charly	ML

CourseType

course	type
AI	elective
DB	core
ML	core

Who took all core courses in RA with relational division?

$$\text{Studies} \div \pi_{\text{course}}\left(\sigma_{\text{type}='\text{core}'}\text{CourseType}\right)$$

How to write R÷S in Primitive RA? (×,−, π)

$$R(X,Y) \div S(Y)$$

$$R \div S = Q$$

X	Y
a	0
a	1
a	2
b	1

Y
1
2

?

How to write R÷S in Primitive RA? (×,–, π)

$$R(X,Y) \div S(Y)$$

?

$R \div S = Q$

X	Y
a	0
a	1
a	2
b	1
b	2

Y
1
2

X
a

1

2

3

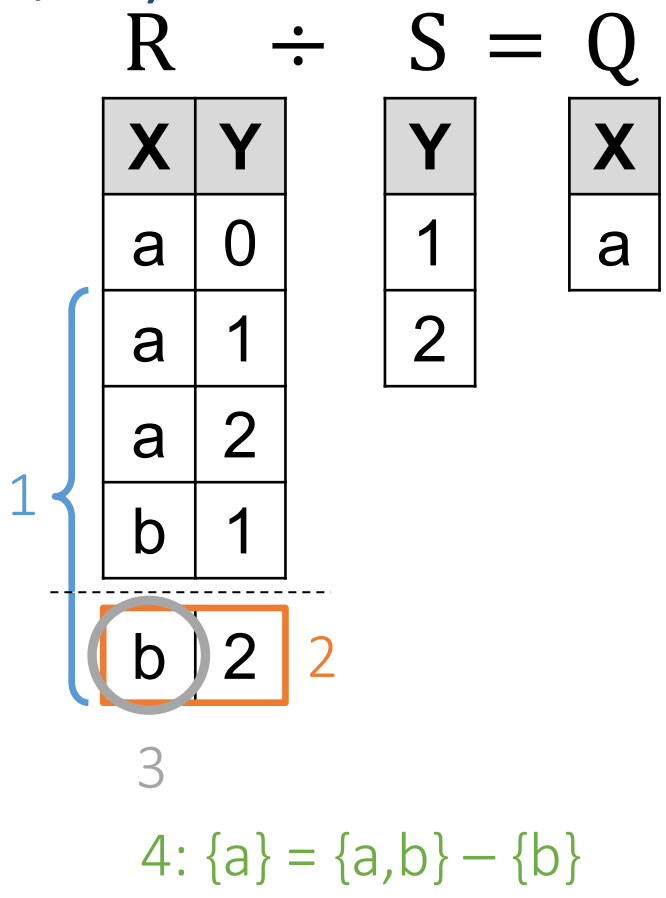
4: {a} = {a,b} – {b}

How to write R÷S in Primitive RA? (×,–, π)

$$R(X,Y) \div S(Y)$$

$$\pi_X R \times S$$

Each X of R w/ each Y of S



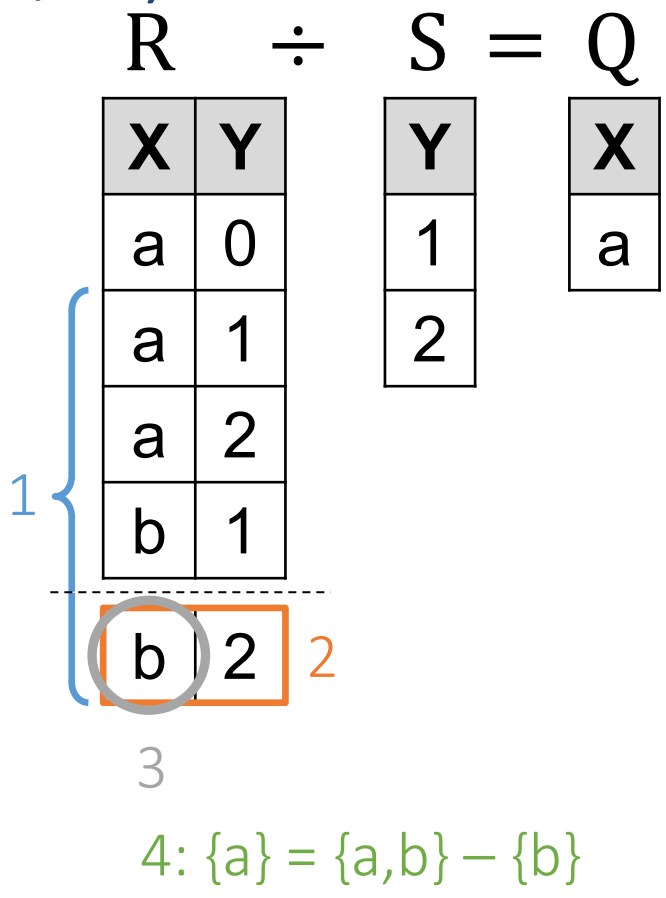
How to write R÷S in Primitive RA? (×,–, π)

$$R(X,Y) \div S(Y)$$

$$(\pi_X R \times S) - R$$

Each X of R w/ each Y of S

(X,Y) s.t. X in R, Y in S, but (X,Y) not in R



How to write R÷S in Primitive RA? (×,–, π)

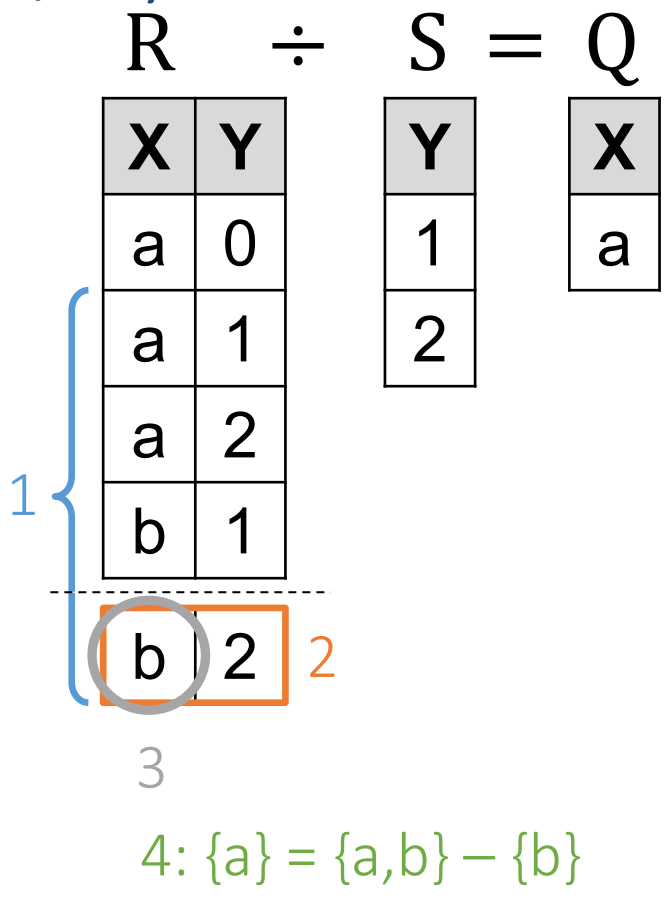
$$R(X,Y) \div S(Y)$$

$$\pi_X\Big(\big(\boxed{\pi_X R \times S}\big) - R\Big)$$

Each X of R w/ each Y of S

(X,Y) s.t. X in R, Y in S, but (X,Y) not in R

Xs in R where for some Y in S, (X,Y) is not in R



How to write R÷S in Primitive RA? (×,–, π)

$$R(X,Y) \div S(Y)$$

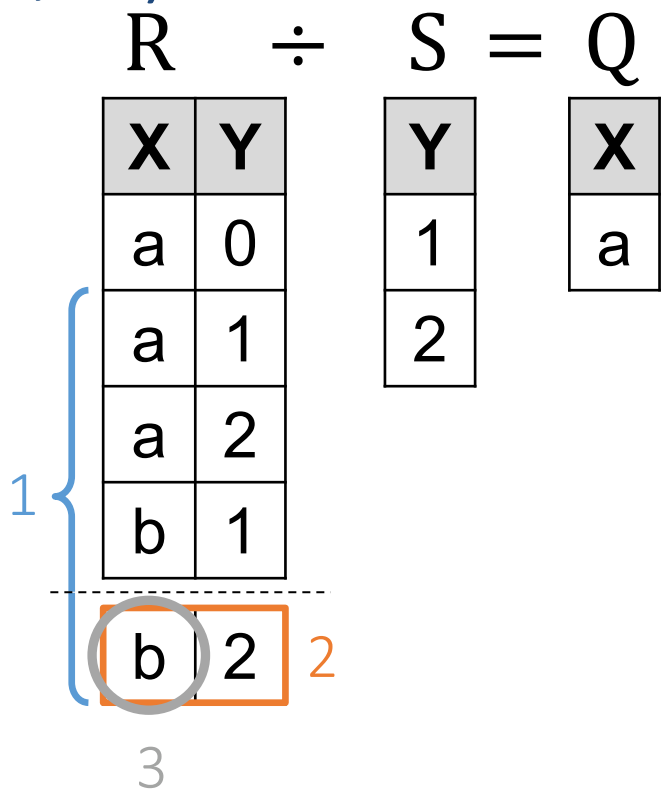
$$\pi_X R - \pi_X\Big(\big(\pi_X R \times S\big) - R\Big)$$

Each X of R w/ each Y of S

(X,Y) s.t. X in R, Y in S, but (X,Y) not in R

Xs in R where for some Y in S, (X,Y) is not in R

R÷S



What if S=∅?

$$R(X,Y) \div S(Y)$$

$$R \div S = Q$$

X	Y
a	0
a	1
a	2
b	1

Y

?

What if S=∅?

$$R(X,Y) \div S(Y)$$

$$\pi_X R - \pi_X\big((\pi_X R \times S) - R\big)$$

R ÷ S = Q

X	Y
a	0
a	1
a	2
b	1

Y

X
a
b

Recall: {t[X] | t[X,Y]∈R for all s[Y]∈S}

Now you see why we needed the safety condition "$T \subseteq \pi_X R$" when defining "R ÷ S as the largest relation T(X) s.t. S × T ⊆ R"

R÷S in Primitive RA vs. RC

$$R(X,Y) \div S(Y)$$

In RA:

$$\pi_X R - \pi_X\big((\pi_X R \times S) - R\big)$$

In DRC: ?

$R \div S = Q$

X	Y
a	0
a	1
a	2
b	1
b	2

Y
1
2

X
a

R÷S in Primitive RA vs. RC

$$R(X,Y) \div S(Y)$$

R ÷ S = Q

X	Y
a	0
a	1
a	2
b	1
b	2

Y
1
2

X
a

In RA:

$$\pi_X R - \pi_X\big((\pi_X R \times S) - R\big)$$

In DRC:

$$\{ X \mid \exists Z.[R(X,Z)] \wedge \quad ? \quad \}$$

X is "guarded": safe and thus domain independent

R÷S in Primitive RA vs. RC

$$R(X,Y) \div S(Y)$$

R ÷ S = Q

X	Y
a	0
a	1
a	2
b	1
b	2

Y
1
2

X
a

In RA:

$$\pi_X R - \pi_X\big((\pi_X R \times S) - R\big)$$

In DRC: what if S(Y)=∅ ?

$$\{ X \mid \exists Z.[R(X,Z)] \wedge \forall Y.[S(Y) \rightarrow R(X,Y)] \}$$

R÷S in Primitive RA vs. RC

$$R(X,Y) \div S(Y)$$

R ÷ S = Q

X	Y
a	0
a	1
a	2
b	1
b	2

Y
1
2

X
a

In RA:

$$\pi_X R - \pi_X\big((\pi_X R \times S) - R\big)$$

In DRC: what if S(Y)=∅ ?

$$\{ X \mid \exists Z.[R(X,Z)] \wedge \forall Y.[S(Y) \rightarrow R(X,Y)] \}$$

? without universal quantification

R÷S in Primitive RA vs. RC

$$R(X,Y) \div S(Y)$$

R ÷ S = Q

X	Y
a	0
a	1
a	2
b	1
b	2

Y
1
2

X
a

In RA:

$$\pi_X R - \pi_X\big((\pi_X R \times S) - R\big)$$

In DRC: what if S(Y)=∅ ?

$$\{ X \mid \exists Z.[R(X,Z)] \wedge \forall Y.[S(Y) \rightarrow R(X,Y)] \}$$

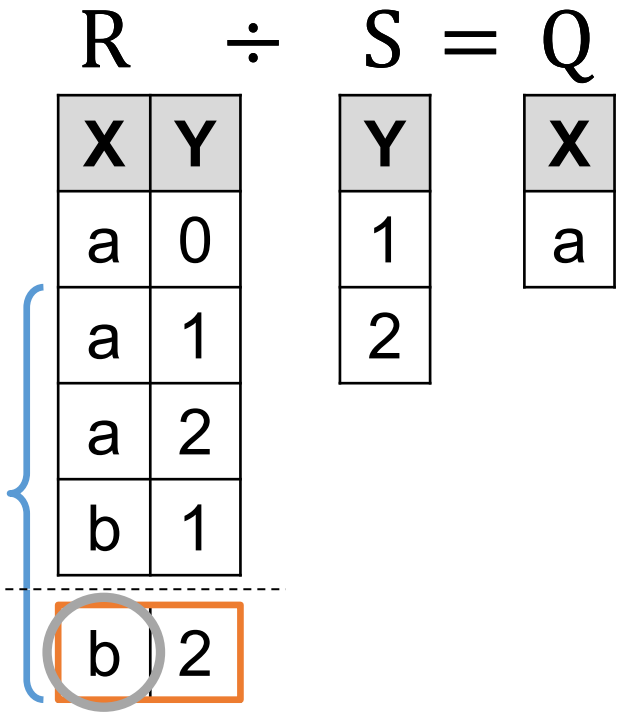
$$\{ X \mid \exists Z.[R(X,Z)] \wedge \nexists Y.[S(Y) \wedge \neg R(X,Y)] \}$$

In TRC:

?

R÷S in Primitive RA vs. RC

$$R(X,Y) \div S(Y)$$



In RA:

$$\pi_X R - \pi_X\big((\pi_X R \times S) - R\big)$$

In DRC: what if S(Y)=∅ ?

$$\{ X \mid \exists Z.[R(X,Z)] \wedge \forall Y.[S(Y) \rightarrow R(X,Y)] \}$$

$$\{ X \mid \exists Z.[R(X,Z)] \wedge \nexists Y.[S(Y) \wedge \neg R(X,Y)] \}$$

∀ ∃ ¬ in SQL

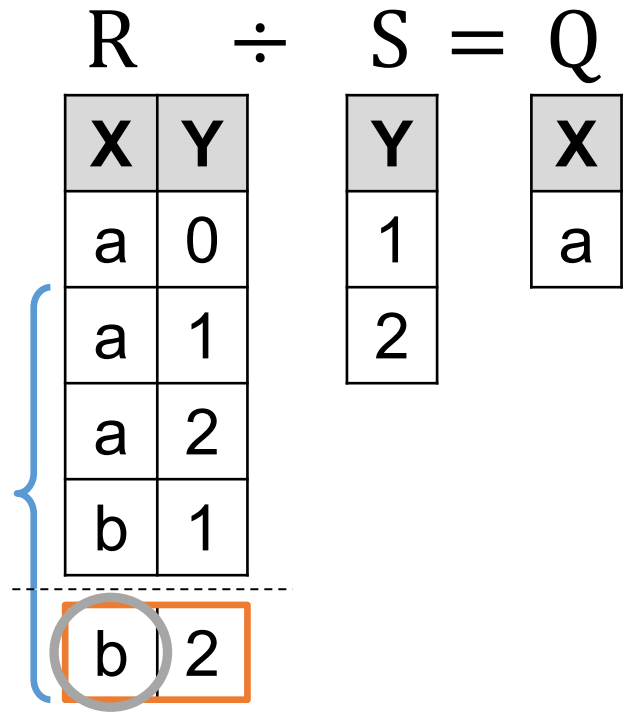
In TRC:

$$\{ r.A \mid \exists r \in R.[\nexists s \in S.[\nexists r_2 \in R.[r_2.B = s.B \wedge r_2.A = r.A)]] \}$$

R÷S in Primitive RA vs. RC

In SQL

```
SELECT DISTINCT R.A
FROM R
WHERE not exists (
        SELECT *
        FROM S
        WHERE not exists (
                SELECT *
                FROM R AS R2
                WHERE R2.B=S.B
                AND R2.A=R.A))
```



In TRC:

$$\{ r.A \mid \exists r \in R.[\, \nexists s \in S.[\nexists r_2 \in R.[r_2.B = s.B \wedge r_2.A = r.A)] \,] \}$$

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:

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

becomes

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