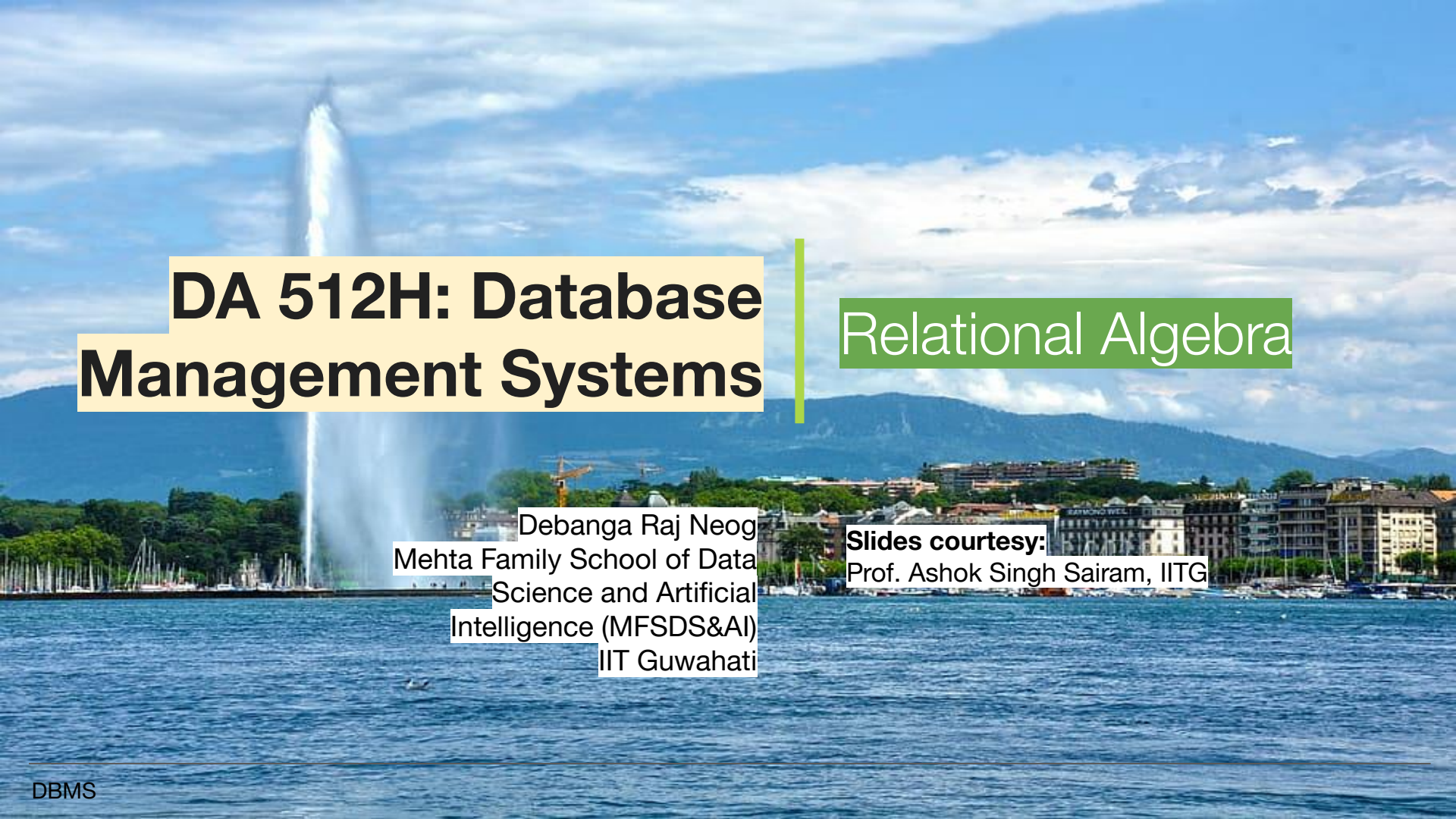


DA 512H: Database Management Systems

Relational Algebra

Debanga Raj Neog
Mehta Family School of Data Science and Artificial Intelligence (MFSDS&AI)
IIT Guwahati

Slides courtesy:
Prof. Ashok Singh Sairam, IITG

Lecture outline

- Relational algebra Preliminaries
- Relational algebra Operators
  - Selection
  - Projection
  - Union, Intersection
  - Set difference
  - Cartesian Product

Why Relational Algebra matter?

- An essential topic to understand how query processing and optimization work
  - What happens when an SQL is issued to a database?
- Help you master the skills to quickly learn a new query language
  - How to quickly learn XML QL and MongoDB QL?

Relational Query Languages

- Query languages allow the manipulation and retrieval of data from a database
- Two mathematical Query Languages form the basis for “real” languages (e.g. SQL),
- Relational Algebra
  - Procedural query language (step-by-step procedure)
  - used to represent execution plans
- Relational Calculus
  - Non-procedural (declarative) query language
  - Describe **what** you want, rather than **how** to compute it
  - Foundation for SQL

What is an algebra?

- What is an “Algebra”?
  - Set of operands and a set of composable operations that they are “closed” under
- Examples:
  - Boolean algebra - operands are the logical values True and False, and operations include AND(), OR(), NOT(), etc.
  - Integer algebra - operands are the set of integers, operands include ADD(), SUB(), MUL(), NEG(), etc. many of which have special infix operator symbols (+,-,*,-)
- Relational Algebra - “operands” are relations, what are the operators?

Preliminaries: Results of a query

- Query is a function over relations

$$Q(R_1,\ldots,R_n) = R_{result}$$

- A query is applied to relation instances, and the result of a query is also a relation instance.
- The schema of the result relation is determined by the input relation and the query
- Because the result of a query is a relation, it can be used as input to another query

Preliminaries: Positional vs. named-field

- Named-field notation:
  - Use field names to refer to fields, R.name
  - Makes queries more readable
- Positional Notation:
  - Refer to fields by position, R[i]
  - easier for formalism
- Both available in SQL

Relational Algebra Operators

σ Sigma
π Pi
ρ Rho

- Core 5 operators
  - Selection (σ) - Selects a subset of rows from relation
  - Projection (π) - Deletes unwanted columns from relation
  - Union (∪) - Tuples in relation 1 and in relation 2
  - Set Difference (-) - Tuples in relation 1, but not in relation 2
  - Cross product (X) - Allows us to combine two relations
- Additional operators
  - Rename (ρ) - Assign names to the results of query; Not essential, but (very!) useful
  - join (⋈) – Combining relations
  - Intersect (∩) – Tuples common in relation 1 and relation 2

Selection

- The selection operator, σ (sigma), specifies the rows to be retained from the input relation
- A selection has the form: $\sigma_{condition}(relation)$, where condition is a Boolean expression
  - Terms in the condition are comparisons between two fields (or a field and a constant)
    - the fields can be referenced by name or position
  - Using one of the comparison operators: $<, \leq, =, \neq, \geq, >$
  - Terms may be connected by $\wedge$ (and), or $\vee$ (or),
  - Terms may be negated using $\neg$ (not)

Selection Example

$\sigma_{birth < 1981}(Customer)$

Customer

sin	firstName	lastName	birth
111	Buffy	Summers	1981
222	Xander	Harris	1981
333	Cordelia	Chase	1980
444	Rupert	Giles	1955
555	Dawn	Summers	1984

sin	firstName	lastName	birth
333	Cordelia	Chase	1980
444	Rupert	Giles	1955

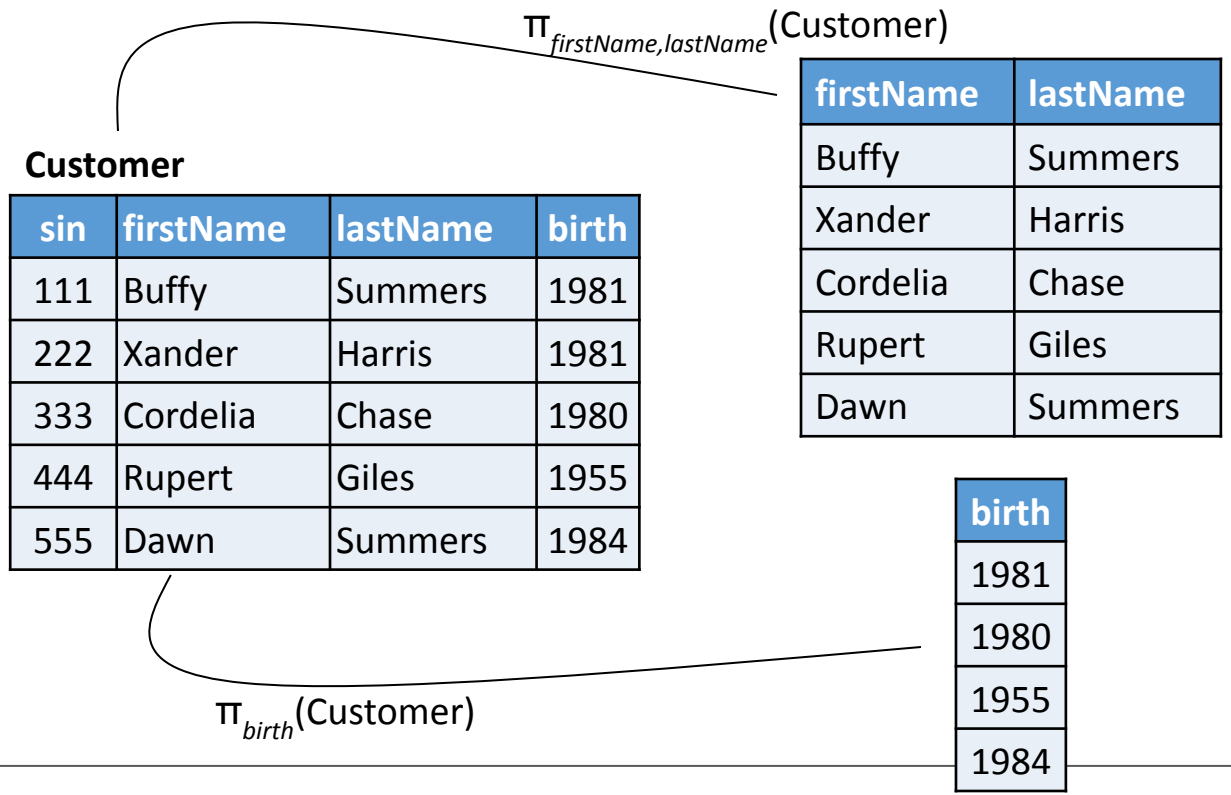
$\sigma_{lastName = "Summers"}(Customer)$

sin	firstName	lastName	birth
111	Buffy	Summers	1981
555	Dawn	Summers	1984

Projection

- The projection operator, π (pi), specifies the columns to be retained from the input relation
- A selection has the form: $\pi_{columns}(relation)$
  - Where *columns* is a comma separated list of column names
  - The list contains the names of the columns to be retained in the result relation

Projection Example



Selection and Projection Notes

1. **Which of the following eliminate duplicates?**

   (a) Selection (b) Projection (c) Both

1. **Which of the following require only one input relation?**

   (a) Selection (b) Projection (c) Both

1. **True or False?**

   “The schema of the result of a selection is the same as the schema of the input relation”

Selection and Projection Notes

- Projection eliminate duplicates when two rows contain same info only differ by primary key
  - Since relations are sets
- Both operations require one input relation
- The schema of the result of a selection is *the same as* the schema of the input relation
- The schema of the result of a projection contains just those attributes in the projection list

Composing Selection and Projection

- List the *sin* and *firstname* of all customers born before 1982 and last name is Summers.

Customer

sin	firstName	lastName	birth
111	Buffy	Summers	1981
222	Xander	Harris	1981
333	Cordelia	Chase	1980
444	Rupert	Giles	1955
555	Dawn	Summers	1984

Composing Selection and Projection

- List the *sin* and *firstname* of all customers born before 1982 and last name is Summers.

$$\pi_{sin,\ firstName}(\sigma_{birth < 1982\ \wedge\ lastName = "Summers"}(Customer))$$

Customer

sin	firstName	lastName	birth
111	Buffy	Summers	1981
222	Xander	Harris	1981
333	Cordelia	Chase	1980
444	Rupert	Giles	1955
555	Dawn	Summers	1984

intermediate relation

sin	firstName	lastName	birth
111	Buffy	Summers	1981

sin	firstName
111	Buffy

Composing Selection and Projection

Customer

sin	firstName	lastName	birth
111	Buffy	Summers	1981
222	Xander	Harris	1981
333	Cordelia	Chase	1980
444	Rupert	Giles	1955
555	Dawn	Summers	1984

$$\pi_{columns}(\sigma_{condition}(R)) = \sigma_{condition}(\pi_{columns}(R))$$

Is it True?

Only when *condition* is on a subset of *columns*

Composing Selection and Projection

$\pi_{birth}(\sigma_{birth < 1981}(Customer))$

birth
1980
1955

Customer

sin	firstName	lastName	birth
111	Buffy	Summers	1981
222	Xander	Harris	1981
333	Cordelia	Chase	1980
444	Rupert	Giles	1955
555	Dawn	Summers	1984

$\pi_{columns}(\sigma_{condition}(R)) = \sigma_{condition}(\pi_{columns}(R))$

birth
1980
1955

$\sigma_{birth < 1981}(\pi_{birth}(Customer))$

Relational Algebra Operators

σ Sigma
π Pi
ρ Rho

- Core 5 operators
  - Selection (σ) - Selects a subset of rows from relation
  - Projection (π) - Deletes unwanted columns from relation
  - **Union (∪) - Tuples in relation 1 and in relation 2**
  - **Set Difference (-) - Tuples in relation 1, but not in relation 2**
  - Cross product (X) - Allows us to combine two relations
- Additional operators
  - Rename (ρ) - Assign names to the results of query; Not essential, but (very!) useful
  - join (⋈) – Combining relations
  - **Intersect (∩) – Tuples common in relation 1 and relation 2**

Set Operations Review

A = {1, 3, 6} **B = {1, 2, 5, 6}**

Union (∪)	A ∪ B ≡ B ∪ A	A ∪ B = {1, 2, 3, 5, 6}	
Intersection(∩)	A ∩ B ≡ B ∩ A	A ∩ B = {1, 6}	
Set Difference(−)	A − B ≠ B − A	A − B = {3}	B − A = {2, 5}

Consider Union of Relations

$A \cup B = R_{result}$

- What will be the result?

A

sin	firstName	lastName
111	Buffy	Summers
222	Xander	Harris
333	Cordelia	Chase
444	Rupert	Giles
555	Dawn	Summers

B

sin	firstName	lastName
208	Clark	Kent
111	Buffy	Summers
412	Carol	Danvers

Consider Union of Relations

$A \cup B = R_{result}$

- What will be the result?

A

sin	firstName	lastName
111	Buffy	Summers
222	Xander	Harris
333	Cordelia	Chase
444	Rupert	Giles
555	Dawn	Summers

B

sin	FirstName	LastName
208	Clark	Kent
111	Buffy	Summers
412	Carol	Danvers

Consider Union of Relations

$A \cup B = R_{result}$

- What will be the result?

A

sin	Age	firstName	lastName
111	22	Buffy	Summers
222	31	Xander	Harris
333	40	Cordelia	Chase
444	28	Rupert	Giles
555	32	Dawn	Summers

B

sin	FirstName	LastName
208	Clark	Kent
111	Buffy	Summers
412	Carol	Danvers

Union

- $A \cup B = R_{result}$
  - Returns a relation instance (R_{result}) containing tuples that occur in either relation instance A or relation instance B (or both)
  - Schema of R_{result} identical to schema of A
- A and B must be union compatible
  - Same number of fields
  - corresponding field i in each schema have the same type

    [field names need not be identical]
  - Assume R_{result} inherit the names from A

    A(age int) $\cup$ **B**(num int) = ?

Union

A

sin	firstName	lastName
111	Buffy	Summers
222	Xander	Harris
333	Cordelia	Chase
444	Rupert	Giles
555	Dawn	Summers

B

sin	firstName	lastName
208	Clark	Kent
111	Buffy	Summers
412	Carol	Danvers

A ∪ B

sin	firstName	lastName
111	Buffy	Summers
222	Xander	Harris
333	Cordelia	Chase
444	Rupert	Giles
555	Dawn	Summers
208	Clark	Kent
412	Carol	Danvers

Intersection

- $A \cap B = R_{result}$
  - Returns a relation instance (R_{result}) containing tuples that occur both in relation instance A and relation instance B
  - Schema of R_{result} identical to schema of A
  - *A* and *B* must be union compatible

Intersection

A

sin	firstName	lastName
111	Buffy	Summers
222	Xander	Harris
333	Cordelia	Chase
444	Rupert	Giles
555	Dawn	Summers

B

sin	firstName	lastName
208	Clark	Kent
111	Buffy	Summers
412	Carol	Danvers

A ∩ B

sin	firstName	lastName
111	Buffy	Summers

Union Compatible Relations

Intersection of the Employee and Customer relations

Customer

sin	firstName	lastName	birth
111	Buffy	Summers	1981
222	Xander	Harris	1981
333	Cordelia	Chase	1980
444	Rupert	Giles	1955
555	Dawn	Summers	1984

Employee

sin	firstName	lastName	salary
208	Clark	Kent	80000.55
111	Buffy	Summers	22000.78
412	Carol	Danvers	64000.00

Is it union compatible?

Union Compatible Relations

Intersection of the Employee and Customer relations

Customer

sin	firstName	lastName	birth
111	Buffy	Summers	1981
222	Xander	Harris	1981
333	Cordelia	Chase	1980
444	Rupert	Giles	1955
555	Dawn	Summers	1984

Employee

sin	firstName	lastName	salary
208	Clark	Kent	80000.55
111	Buffy	Summers	22000.78
412	Carol	Danvers	64000.00

The two relations are not union compatible as birth is a DATE and salary is a REAL

How to then perform intersection?

Union Compatible Relations

Intersection of the Employee and Customer relations

Customer

sin	firstName	lastName	birth
111	Buffy	Summers	1981
222	Xander	Harris	1981
333	Cordelia	Chase	1980
444	Rupert	Giles	1955
555	Dawn	Summers	1984

Employee

sin	firstName	lastName	salary
208	Clark	Kent	80000.55
111	Buffy	Summers	22000.78
412	Carol	Danvers	64000.00

The two relations are not union compatible as birth is a DATE and salary is a REAL

We can carry out preliminary operations to make the relations union compatible

$\pi_{sin,\ firstName,\ lastName}(\text{Customer}) \cap \pi_{sin,\ firstName,\ lastName}(\text{Employee})$

Set Difference

- $A - B = R_{result}$
  - Returns a relation instance (R_{result}) containing tuples that occur in relation instance A but not in relation instance B
  - Schema of R_{result} identical to schema of A
  - *A* and *B* must be union compatible

Set Difference

A

sin	firstName	lastName
111	Buffy	Summers
222	Xander	Harris
333	Cordelia	Chase
444	Rupert	Giles
555	Dawn	Summers

A − B

B

sin	firstName	lastName
208	Clark	Kent
111	Buffy	Summers
412	Carol	Danvers

B − A

Set Difference

A

sin	firstName	lastName
111	Buffy	Summers
222	Xander	Harris
333	Cordelia	Chase
444	Rupert	Giles
555	Dawn	Summers

A − B

sin	firstName	lastName
222	Xander	Harris
333	Cordelia	Chase
444	Rupert	Giles
555	Dawn	Summers

B

sin	firstName	lastName
208	Clark	Kent
111	Buffy	Summers
412	Carol	Danvers

B − A

sin	firstName	lastName
208	Clark	Kent
412	Carol	Danvers

Relational Algebra Operators

σ Sigma
π Pi
ρ Rho

- Core 5 operators
  - Selection (σ) - Selects a subset of rows from relation
  - Projection (π) - Deletes unwanted columns from relation
  - Union (∪) - Tuples in relation 1 and in relation 2
  - Set Difference (-) - Tuples in relation 1, but not in relation 2
  - **Cross product (X) - Allows us to combine two relations**
- Additional operators
  - **Rename (ρ) - Assign names to the results of query; Not essential, but (very!) useful**
  - join (⋈) – Combining relations
  - Intersect (∩) – Tuples common in relation 1 and relation 2

Cartesian Product

$$A(a_1, \ldots, a_m) \times B(a_{m+1}, \ldots, a_n) = R_{result}(a_1, \ldots, a_m, a_{m+1}, \ldots, a_n)$$

- Each row of A paired with all rows of B
  - Result schema concats A and B's fields
  - Names are inherited if possible (i.e. if not duplicated)
    - If two field names are the same (i.e., a *naming conflict* occurs) and the affected columns are referred to by position
  - If *A* contains *m* records, and *B* contains *n* records, the result relation will contain *m* * *n* records

Cartesian Product

$\sigma_{lastName = "Summers"}$(Customer)

sin	firstName	lastName	birth
111	Buffy	Summers	1981
555	Dawn	Summers	1984

Account

acc	type	balance	sin
01	CHQ	2101.76	111
02	SAV	11300.03	333
03	CHQ	20621.00	444

$\sigma_{lastName = "Summers"}$(Customer) × Account

1	firstName	lastName	birth	acc	type	balance	8
111	Buffy	Summers	1981	01	CHQ	2101.76	111
111	Buffy	Summers	1981	02	SAV	11300.03	333
111	Buffy	Summers	1981	03	CHQ	20621.00	444
555	Dawn	Summers	1984	01	CHQ	2101.76	111
555	Dawn	Summers	1984	02	SAV	11300.03	333
555	Dawn	Summers	1984	03	CHQ	20621.00	444

Renaming

$$\rho(C(\bar{F}), E)$$

- takes a relational algebra expression E and returns an instance of a (new) relation called R
- R contains same tuples as result of E
- Field names in R same as in E, except some fields are renamed
- $\bar{F}$ list of terms that have been renamed

oldname → newname

When can it be useful?

Renaming

$$\rho(C(\bar{F}), E)$$

- takes a relational algebra expression E and returns an instance of a (new) relation called R
- R contains same tuples as result of E
- Field names in R same as in E, except some fields are renamed
- $\bar{F}$ list of terms that have been renamed

  oldname → newname
- Useful to resolve naming conflicts can arise during an operation such as Cartesian product
- Example

  $\rho(C(1 \rightarrow sin1, 8 \rightarrow sin2), A \times B)$

Exercise 1

- *Conflict*: Both S1 and R1 have a field called *sid*.a

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	22	101	10/ 10/ 96
22	dustin	7	45.0	58	103	11/ 12/ 96
31	lubber	8	55.5	22	101	10/ 10/ 96
31	lubber	8	55.5	58	103	11/ 12/ 96
58	rusty	10	35.0	22	101	10/ 10/ 96
58	rusty	10	35.0	58	103	11/ 12/ 96

S1 R1

Exercise 1

- *Conflict*: Both S1 and R1 have a field called *sid*.a

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	22	101	10/ 10/ 96
22	dustin	7	45.0	58	103	11/ 12/ 96
31	lubber	8	55.5	22	101	10/ 10/ 96
31	lubber	8	55.5	58	103	11/ 12/ 96
58	rusty	10	35.0	22	101	10/ 10/ 96
58	rusty	10	35.0	58	103	11/ 12/ 96

S1 R1

- **Renaming operator**: $\rho\,(C(1 \to sid1, 5 \to sid2), S1 \times R1)$

Relational Algebra Exercises

- **Student** (rollNo, lastName, firstName, cpi)
  - 101, Jordan, Michael, 3.8
- **Courses** (cID, dept, cName, term, instructor)
  - DA214, Maths, DBMS, Fall 2023, Debanga
  - CS348, CS, Networks, Fall 2022, Tom
- **Enroll** (rollNo, cID, grade)
  - 101, DA214, 9
  - 101, CS348, 5

1. rollNo of all students who have earned some grade over 8 and some grade below 6.

Relational Algebra Exercises

- **Student** (rollNo, lastName, firstName, cpi)
  - 101, Jordan, Michael, 3.8
- **Courses** (cID, dept, cName, term, instructor)
  - DA214, Maths, DBMS, Fall 2023, Debanga
  - CS348, CS, Networks, Fall 2022, Tom
- **Enroll** (rollNo, cID, grade)
  - 101, DA214, 9
  - 101, CS348, 5

1. rollNo of all students who have earned some grade over 8 and some grade below 6.

$$\pi_{rollNo}(\sigma_{grade > 8}(\text{Enroll})) \cap \pi_{rollNo}(\sigma_{grade < 6}(\text{Enroll}))$$

- **Courses** (cID, dept, cName, term, instructor)
  - DA214, DSAI, DBMS, Fall 2023, Debanga
  - CS348, CS, Networks, Fall 2022, Tom

- **Enroll** (rollNo, cID, grade)
  - 101, DA214, 9
  - 101, CS348, 5

2. Roll number of all students who have taken DBMS offered by DSAI department

- **Courses** (cID, dept, cName, term, instructor)
  - DA214, DSAI, DBMS, Fall 2023, Debanga
  - CS348, CS, Networks, Fall 2022, Tom

- **Enroll** (rollNo, cID, grade)
  - 101, DA214, 9
  - 101, CS348, 5

2. Roll number of all students who have taken DBMS offered by DSAI department

$$\pi_{rollNo}(\sigma_{Courses.cID = Enroll.cID \wedge dept = DSAI \wedge cName = DBMS}(\text{Courses x Enroll}))$$

Relational Algebra Operators - review

- Core 5 operators
  - Selection (σ) - Selects a subset of rows from relation
  - Projection (π) - Deletes unwanted columns from relation
  - Union (∪) - Tuples in relation 1 and in relation 2
  - Set Difference (-) - Tuples in relation 1, but not in relation 2
  - Cross product (X) - Allows us to combine two relations

σ Sigma
π Pi
ρ Rho

- Additional operators
  - Rename (ρ) - Assign names to the results of query; Not essential, but (very!) useful
  - **join (⋈) – Combining relations**
  - Intersect (∩) – Tuples common in relation 1 and relation 2

Example

- **Student** (rollNo, lastName, firstName, cpi)
  - 101, Jordan, Michael, 3.8
  - 102, Mathew, Tim, 3.9
- **Courses** (cID, dept, cName)
  - DA214, DSAI, DBMS
  - DA243, DSAI, OPT
  - CS345, CS, DBMS
- **Enroll** (rollNo, cID)
  - 101, DA214
  - 102, CS345

Q. Roll number of all students who have taken DBMS offered by DSAI

Example

- **Student** (rollNo, lastName, firstName, cpi)
  - 101, Jordan, Michael, 3.8
  - 102, Mathew, Tim, 3.9
- **Courses** (cID, dept, cName)
  - DA214, DSAI, DBMS
  - DA243, DSAI, OPT
  - CS345, CS, DBMS
- **Enroll** (rollNo, cID)
  - 101, DA214
  - 102, CS345

Q. Roll number of all students who have taken DBMS offered by DSAI

$$\pi_{rollNo}\left(\sigma_{Courses.cID = Enroll.cID \wedge Courses.dept = DSAI \wedge Courses.cName = DBMS}\left(\textbf{Courses x Enroll}\right)\right)$$

Example

- **Student** (rollNo, lastName, firstName, cpi)
  - 101, Jordan, Michael, 3.8
  - 102, Mathew, Tim, 3.9
- **Courses** (cID, dept, cName)
  - DA214, DSAI, DBMS
  - DA243, DSAI, OPT
  - CS345, CS, DBMS
- **Enroll** (rollNo, cID)
  - 101, DA214
  - 102, CS345

Example selection in SQL:

```
SELECT *
      FROM Courses, Enroll;
```

Example projection in SQL:

```
SELECT Enroll.rollNo
   FROM Courses, Enroll
   WHERE Courses.cID = Enroll.cID AND
Courses.dept = "DSAI" AND
Courses.cName="DBMS";
```

Q. Roll number of all students who have taken DBMS offered by DSAI

$$\Pi_{rollNo}\left(\sigma_{Courses.cID = Enroll.cID \wedge Courses.dept = DSAI \wedge Courses.cName = DBMS}\left(\text{Courses x Enroll}\right)\right)$$

Example

- **Student** (rollNo, lastName, firstName, cpi)
  - 101, Jordan, Michael, 3.8
  - 102, Mathew, Tim, 3.9
- **Courses** (cID, dept, cName)
  - DA214, DSAI, DBMS
  - DA243, DSAI, OPT
  - CS345, CS, DBMS
- **Enroll** (rollNo, cID)
  - 101, DA214
  - 102, CS345

Q. Name of all students who have taken DBMS offered by DSAI department

Example

- **Student** (rollNo, lastName, firstName, cpi)
  - 101, Jordan, Michael, 3.8
  - 102, Mathew, Tim, 3.9
- **Courses** (cID, dept, cName)
  - DA214, DSAI, DBMS
  - DA243, DSAI, OPT
  - CS345, CS, DBMS
- **Enroll** (rollNo, cID)
  - 101, DA214
  - 102, CS345

Cartesian Product:

```
SELECT *
   FROM Courses, Enroll, Student
```

Finding student names:

```
SELECT Student.firstName, Student.lastName
   FROM Courses, Enroll, Student
   WHERE Student.rollNo = Enroll.rollNo AND
Courses.cID = Enroll.cID AND Courses.dept =
"DSAI" AND Courses.cName="DBMS";
```

Q. Name of all students who have taken DBMS offered by DSAI department

$$\Pi_{lastName,\ firstName}\left(\sigma_{Student.rollNo = Enroll.rollNo \wedge Courses.cID = Enroll.cID \wedge Courses.dept = DSAI \wedge Courses.cName = DBMS}(Courses \times Enroll \times Student)\right)$$

Joins

- **Cross Product**: Combine rows from two or more tables
- **Join**: Combine rows from two or more tables, based on a related column between them
- Two Types – Inner join and Outer join
  - Inner join is most widely used and considered as default type
  - Three types of inner join
- Motivation
  - Simplify some queries that require a Cartesian product

(Inner) Joins types

- **Condition (or Theta) Join:** $R \bowtie_\theta S = \sigma_\theta (R \times S)$
  - Allows for arbitrary comparison (<,≤, =, >, ≥) among the attributes of relation R and S
- **Equijoin:** $R \bowtie_\theta S = \sigma_\theta (R \times S)$
  - Join condition θ consists only of equalities
- **Natural Join:** $R \bowtie S = \pi_A (\sigma_\theta (R \times S))$
  - Equijoin on attributes that have the same name in relations R and S

Condition/Theta Join

- **Condition Join**: Most general form of join
  - Cross product followed by a selection
  - θ can be any condition

$$R \bowtie_{\theta} S = \sigma_{\theta}(R \times S)$$

- No projection in this case!
  - Result schema same as cross product
  - Fewer tuples than cross-product, might be able to compute more efficiently

Theta Join Example

Customer

sin	firstName	lastName	birth
111	Buffy	Summers	1981
222	Xander	Harris	1981
333	Cordelia	Chase	1980
444	Rupert	Giles	1955
555	Dawn	Summers	1984

Employee

sin	firstName	lastName	salary
208	Clark	Kent	80000.55
111	Buffy	Summers	22000.78
412	Carol	Danvers	64000.00

Customer $\bowtie_{\text{Customer.sin} < \text{Employee.sin}}$ **Employee**

1	2	3	birth	5	6	7	salary
111	Buffy	Summers	1981	208	Clark	Kent	80000.55
111	Buffy	Summers	1981	412	Carol	Danvers	64000.00
222	Xander	Harris	1981	412	Carol	Danvers	64000.00
333	Cordelia	Chase	1980	412	Carol	Danvers	64000.00

Equi-Joins

$$R \bowtie_{\theta} S = \sigma_{\theta}(R \times S)$$

- A theta join where θ is an equality predicate

Customer

sin	firstName	lastName	birth
111	Buffy	Summers	1981
222	Xander	Harris	1981
333	Cordelia	Chase	1980
444	Rupert	Giles	1955

Employee

sin	firstName	lastName	salary
208	Clark	Kent	80000.55
111	Buffy	Summers	22000.78
396	Dawn	Allen	41000.21

Customer $\bowtie_{\text{Customer.sin = Employee.sin}}$ Employee

1	2	3	birth	5	6	7	salary
111	Buffy	Summers	1981	111	Buffy	Summers	22000.78

Natural Join

$$R \bowtie S$$

- Meaning: $R \bowtie S = \pi_A(\sigma_\theta (R \times S))$
- Where:
  - Selection σ_θ checks equality of all common attributes (i.e., attributes with same names)
  - Projection π_A eliminates duplicate common attributes
- The natural join of two tables with *no fields in common is the Cartesian product*
  - Not the empty set

Natural Join Example

R

A	B	C	D
111	Buffy	Summers	1981
222	Xander	Harris	1981
333	Cordelia	Chase	1980
444	Rupert	Giles	1955

S

A	B	C	E
208	Clark	Kent	80000.55
111	Buffy	Summers	22000.78
396	Dawn	Allen	41000.21

$$R \bowtie S = \pi_{A,B,C,D,E}(\sigma_{R.A=S.A \wedge R.B=S.B \wedge R.C=S.C}(R \times S))$$

A	B	C	D	E
111	Buffy	Summers	1981	22000.78

Relational Algebra Exercises

- **Student (<u>rollNo</u>, lastName, firstName, cpi)**
  - 101, Jordan, Michael, 3.8
- **Course (dept, <u>cid</u>, name, breadth)**
  - Maths, MA354, Algebra, True
- **Offering (<u>oID</u>, dept, cid, term, instructor)**
  - abc, Maths, MA354, Fall 2018, Jiannan
- **Took (<u>rollNo</u>, <u>oID</u>, grade)**
  - 101, abc, 9

The names of all students who have passed a breadth course (grade >= 60 and breadth = True)

$$\pi_{lastName,\ firstName}(\sigma_{breadth = True \wedge grade > 60}(\text{Student} \bowtie \text{Took} \bowtie \text{Offering} \bowtie \text{Course}))$$

Different Plans, Same Results

- Semantic equivalence: results are *always* the same

$$\pi_{name}(\sigma_{cNum=MA354}\ (R \bowtie S))$$

$$\pi_{name}(\sigma_{cNum=MA354}\ (R) \bowtie S)$$

- Are they equivalent?
- One may be more efficient than the other.

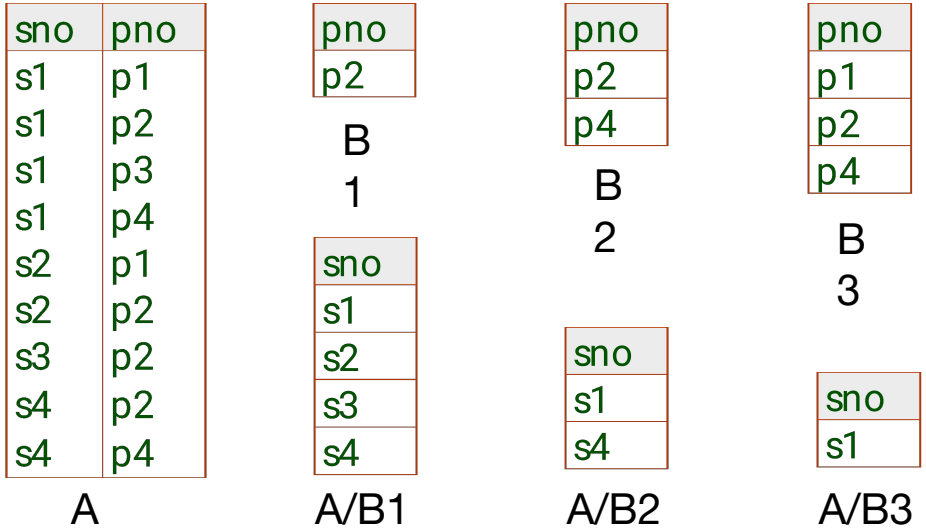
Division

- Not supported as a primitive operator, but useful for expressing queries like:
  *Find suppliers who can supply **all** parts*.
- Let *A* have 2 fields, *x* and *y*; *B* have only field *y*:
  - *A*/*B* =

$$\{\langle x\rangle \mid \exists \langle x, y\rangle \in A \ \ \forall \langle y\rangle \in B\}$$

  - i.e., ***A*/*B* contains all *x* tuples (suppliers) such that for *every* *y* tuple (parts) in *B*, there is an *xy* tuple in *A*.**
  - *Or*: If the set of *y* values (parts) associated with an *x* value (supplier) in *A* contains all *y* values in *B*, the *x* value is in *A*/*B*.
- In general, *x* and *y* can be any lists of fields; *y* is the list of fields in *B*, and *x*$\cup$*y* is the list of fields of *A*.

Examples of Division A/B



More Examples

sid	sname	rating	age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

S3: Sailors

sid	bid	day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

R2: Reserves

bid	bname	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

H1: Boats

Example1

⋈

sid	snam e	rating	age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

S3: Sailors

sid	bid	day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

R2: Reserves

bid	bname	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

H1: Boats

Find names of sailors who've reserved boat #103

Find names of sailors who've reserved boat #103

A Solution: $\pi_{sname}(\sigma_{bid=103}(Reserves \bowtie Sailors))$

Example2

sid	snam e	rating	age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

S3: Sailors

sid	bid	day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

R2: Reserves

bid	bname	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

H1: Boats

Find names of sailors who've reserved a red boat

Find names of sailors who've reserved a red boat

- Information about boat color only available in Boats; so need an extra join:

$$\pi_{sname}((\sigma_{color='red'} Boats) \bowtie Reserves \bowtie Sailors)$$

- A more efficient solution:

$$\pi_{sname}(\pi_{sid}((\pi_{bid}\sigma_{color='red'} Boats) \bowtie Res) \bowtie Sailors)$$

- A query optimizer can find this, given the first solution!

Example3

sid	snam e	rating	age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

S3: Sailors

sid	bid	day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

R2: Reserves

bid	bname	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

H1: Boats

Find sailors who've reserved a red or a green boat

Find sailors who've reserved a red or a green boat

- Can identify all red or green boats, then find sailors who've reserved one of these boats:

$$\rho\ (Tempboats, (\sigma_{color='red' \vee color='green'}, Boats))$$

$$\pi_{sname}(Tempboats \bowtie \text{Re}serves \bowtie Sailors)$$

  - What happens if $\vee$ is replaced by $\wedge$ in this query?

Example4

sid	snam e	rating	age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

S3: Sailors

sid	bid	day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

R2: Reserves

bid	bname	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

H1: Boats

$$\rho\ (Tempred, \pi_{sid}((\sigma_{color='red'} Boats) \bowtie Reserves))$$

$$\rho\ (Tempgreen, \pi_{sid}((\sigma_{color='green'} Boats) \bowtie Reserves))$$

$$\pi_{sname}((Tempred \cap Tempgreen) \bowtie Sailors)$$

Find sailors who've reserved a red and a green boat

Find sailors who've reserved a red and a green boat

Previous approach won't work! Must identify sailors who've reserved red boats, sailors who've reserved green boats, then find the intersection (note that *sid* is a key for Sailors):

$$\rho\ (Tempred, \pi_{sid}((\sigma_{color='red'} Boats) \bowtie Reserves))$$

$$\rho\ (Tempgreen, \pi_{sid}((\sigma_{color='green'} Boats) \bowtie Reserves))$$

$$\pi_{sname}((Tempred \cap Tempgreen) \bowtie Sailors)$$

Example5

sid	snam e	rating	age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

S3: Sailors

sid	bid	day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

R2: Reserves

bid	bname	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

H1: Boats

Find the names of sailors who've reserved all boats

Find the names of sailors who've reserved all boats

- Uses **division**; schemas of the input relations to / must be carefully chosen:

$$\rho\ (Tempsids, (\pi_{sid,bid} \mathrm{Re}serves) / (\pi_{bid} Boats))$$

$$\pi_{sname} (Tempsids \bowtie Sailors)$$

Summary

- Relational Algebra (RA) operators
  - Five core operators: selection, projection, cross-product, union and set difference
  - Additional operators are defined in terms of the core operators: rename, intersection, join
- Theorem: SQL and RA can express exactly the same class of queries
- Multiple RA queries can be equivalent
  - Same semantics but difference performance
  - Form basis for optimizations

RDBMS translate SQL → RA, then optimize RA