Relational Model and Relational Algebra

CMPSCI 445 – Database Systems

Spring 2018

The Relational Model

- The relational data model (Codd, 1970):
  - Data independence: details of physical storage are hidden from users
  - High-level declarative query language
    - say **what** you want, not **how** to compute it.
    - mathematical foundation

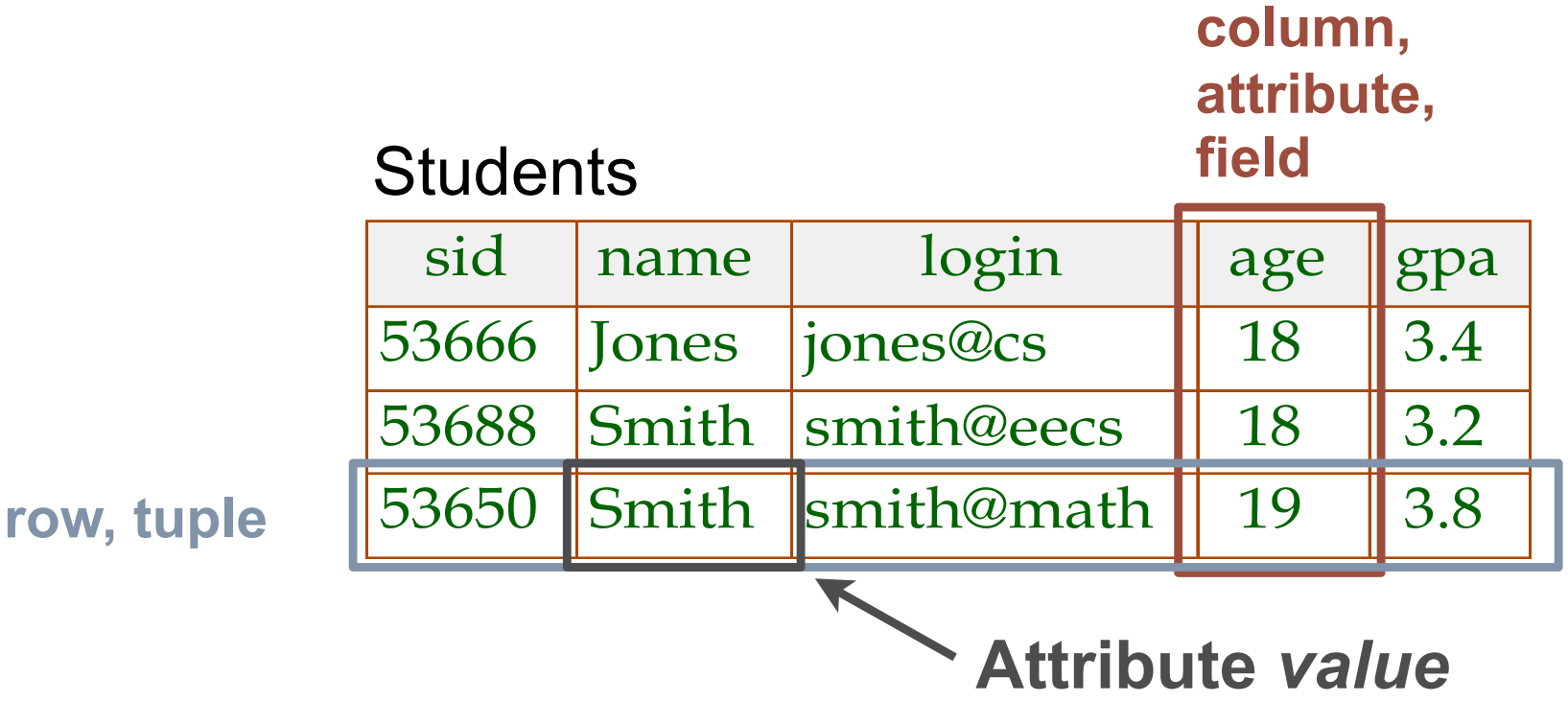
Relational Database: Definitions

- *Relational database:* a collection of *relations*
- *Relation:* made up of 2 parts:
  - *Schema* : specifies name of relation, plus name and type/domain of each column.
  - *Instance* : a *table*, with rows and columns.

Students(*sid*: string, *name*: string, *login*: string, *age*: integer, *gpa*: real).

Restriction: all attributes are of **atomic** type, no nested tables

Relational instances: tables



A relation is a **set** of tuples: no tuple can occur more than once

- Real systems may allow duplicates for efficiency or other reasons – we'll come back to this.

Example Database

STUDENT

sid	name
1	Jill
2	Bo
3	Maya

Takes

sid	cid
1	445
1	483
3	435

COURSE

cid	title	sem
445	DB	F08
483	AI	S08
435	Arch	F08

PROFESSOR

fid	name
1	Diao
2	Saul
8	Weems

Teaches

fid	cid
1	445
2	483
8	435

Relational Query Languages

- ***Query languages**:* Allow the manipulation and retrieval of data from a database.
- DB query languages **!=** programming languages
  - not expected to be “Turing complete”.
  - not intended to be used for complex calculations.
  - support easy, efficient access to large data sets.

Query language preliminaries

Query Q: $R_1..R_n \rightarrow R'$

- A query is applied to one or more *relation instances*
- The result of a query is a relation instance.
- Input and output schema:
  - *Schema* of input relations for a query are fixed.
  - The schema for the *result* of a given query is also fixed: determined by definition of query language constructs.

What is an “Algebra” ?

- Mathematical system consisting of:
  - ***Operands*** --- variables or values from which new values can be constructed.
  - ***Operators*** --- symbols denoting procedures that construct new values from given values.

What is the Relational Algebra?

- An algebra whose **operands** are relations or variables that represent relations.
- **Operators** are designed to do the most common things that we need to do with relations in a database.
  - The result is an algebra that can be used as a ***query language*** for relations.

Relational Algebra

- Operates on relations, i.e. *sets*
  - Things are a bit different on *bags (i.e. multi-sets)*
- Five basic operators:
  - Union: $\cup$
  - Difference: -
  - Selection: σ
  - Projection: Π
  - Cartesian Product: $\times$
- Derived or auxiliary operators:
  - Intersection, complement
  - Joins (natural, equi-join, theta join)
  - Renaming: ρ

1. Union and 2. Difference

R1

sid	name
1	Jill
2	Bo
3	Maya

R2

sid	name
1	Jill
4	Bob

R1 ∪ R2

sid	name
1	Jill
2	Bo
3	Maya
4	Bob

R1 – R2

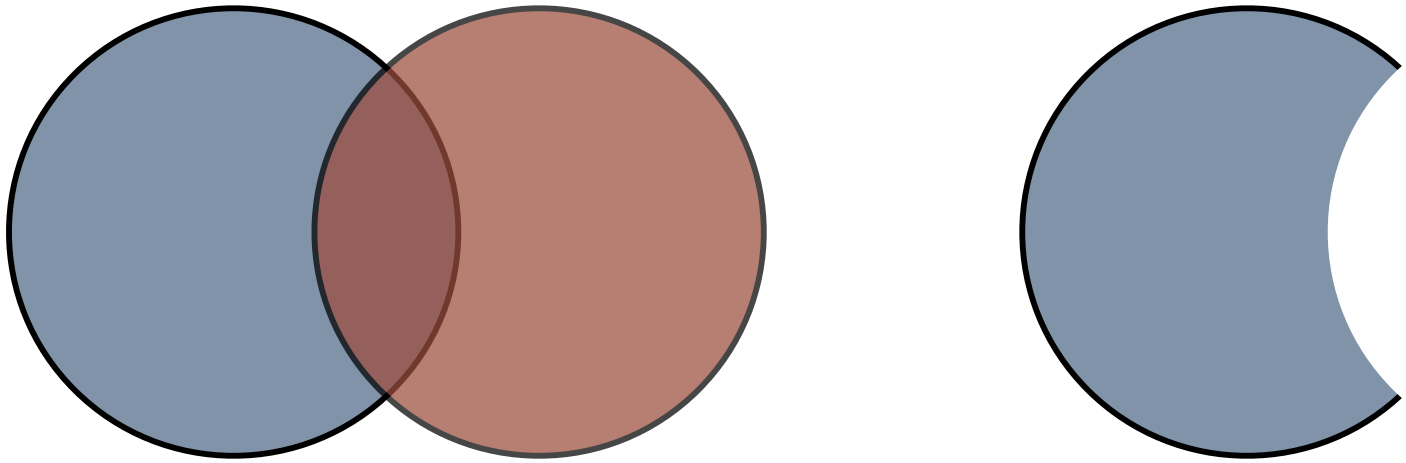
sid	name
2	Bo
3	Maya

i-clicker #1

- Given the following relational tables:
  - P(name,id,color)
  - R(name,id,color)
  - S(id,color)
  - T(name,id)
- Which of the following relational algebra expressions is valid?
  A. R - S
  B. S U T
  C. R U T
  D. P - R

What about Intersection ?

- It is a derived operator
- $R_1 \cap R_2 = R_1 - (R_1 - R_2)$
- Also expressed as a join (will see later)



R_1 R_2 $R_1 - R_2$

3. Selection

- Returns all tuples which satisfy a condition
- Notation: $\sigma_c(R)$
- Examples

  $\sigma_{\text{CID > 400}}$ **(Course)**

  $\sigma_{\text{title = "AI"}}$ **(Course)**

Course

cid	title	sem
445	DB	F08
483	AI	S08
435	Arch	F08

- The condition c can be =, <, ≤, >, ≥, <>

i-clicker #2

- How many rows and columns are in the result table of $\sigma_{age > 30}$ **(Sailors)**
  - A. 4 rows, 1 column
  - B. 7 rows, 1 column
  - C. 7 rows, 3 columns
  - D. 7 rows, 4 columns
  - E. 11 rows, 4 columns

Sailors

sid	sname	rating	age
29	brutus	1	33
85	art	3	25.5
95	bob	3	63.5
96	frodo	3	25.5
22	dustin	7	45
64	horatio	7	35
31	lubber	8	55.5
32	andy	8	25.5
74	horatio	9	35
58	rusty	10	35
71	zorba	10	16

Answer on next slide

i-clicker #2

- How many rows and columns are in the result table of $\sigma_{age > 30}$ **(Sailors)**

  A. 4 rows, 1 column
  B. 7 rows, 1 column
  C. 7 rows, 3 columns
  D. 7 rows, 4 columns
  E. 11 rows, 4 columns

Result

sid	sname	rating	age
29	brutus	1	33
95	bob	3	63.5
22	dustin	7	45
64	horatio	7	35
31	lubber	8	55.5
74	horatio	9	35
58	rusty	10	35

4. Projection

- Eliminates columns, then removes duplicates
- Notation: $\Pi_{A1,\ldots,An}(R)$
- Example: project cid and name

  $\Pi_{\textbf{cid, name}}$ **(Course)**

  Output schema: **Answer(cid, name)**

Course

cid	name	sem
445	DB	F08
483	AI	S08
445	DB	S08

Π →

Answer

cid	name
445	DB
483	AI

i-clicker #3

- How many rows and columns are in the result table of: Π_{rating}**(Sailors)**
  A. 4 rows, 1 column
  B. 4 rows, 4 columns
  C. 2 rows, 1 columns
  D. 2 rows, 4 columns

Sailors

sid	sname	rating	age
29	brutus	1	33
85	art	3	25.5
95	bob	3	63.5
96	frodo	3	25.5

Answer on next slide

i-clicker #3

- How many rows and columns are in the result table of: Π_{rating}**(Sailors)**

  A. 4 rows, 1 column

  B. 4 rows, 4 columns

  C. 2 rows, 1 columns

  D. 2 rows, 4 columns

Result

rating
1
3

5. Cartesian Product

- Each tuple in R_1 with each tuple in R_2
- Notation: $R_1 \times R_2$
- Very rare in practice; mainly used to express joins

Also called “Cross Product”

Cartesian Product

Student

sid	name
1	Jill
2	Bo

Takes

sid	cid
1	445
1	483
3	435

Student × Takes

sid	name	sid	cid
1	Jill	1	445
1	Jill	1	483
1	Jill	3	435
2	Bo	1	445
2	Bo	1	483
2	Bo	3	435

i-clicker #4

- How many rows and columns are in the result table of: $\sigma_{cid = 445}$ (Student x Takes)
  A. 4 rows, 1 column
  B. 2 rows, 4 columns
  C. 3 rows, 4 columns
  D. 4 rows, 4 columns
  E. 6 rows, 6 columns

Student

sid	name
1	Jill
2	Bo

Takes

sid	cid
1	445
1	483
3	435

Answer on next slide

i-clicker #4

- How many rows and columns are in the result table of: $\sigma_{cid = 445}$ (Student x Takes)

  A. 4 rows, 1 column

  B. 2 rows, 4 columns

  C. 3 rows, 4 columns

  D. 4 rows, 4 columns

  E. 6 rows, 6 columns

Student × Takes

sid	name	sid	cid
1	Jill	1	445
2	Bo	1	445

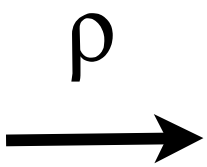
Renaming

- Changes the **schema**, not the **instance**
- Notation: $\rho_{B1,\ldots,Bn}(R)$
- Example:

$\rho_{\textbf{courseID, cname, term}}$ **(Course)**

Course

cid	name	sem
445	DB	F08
483	AI	S08
445	DB	S08



courseID	cname	term
445	DB	F08
483	AI	S08
445	DB	S08

Natural Join

- Notation: $R_1 \bowtie R_2$
- Definition: $R_1 \bowtie R_2 = \Pi_A(\sigma_C(R_1 \times R_2))$
- Where:
  - The selection σ_C checks equality of **all common attributes**
  - The projection eliminates one of the **duplicate common attributes**

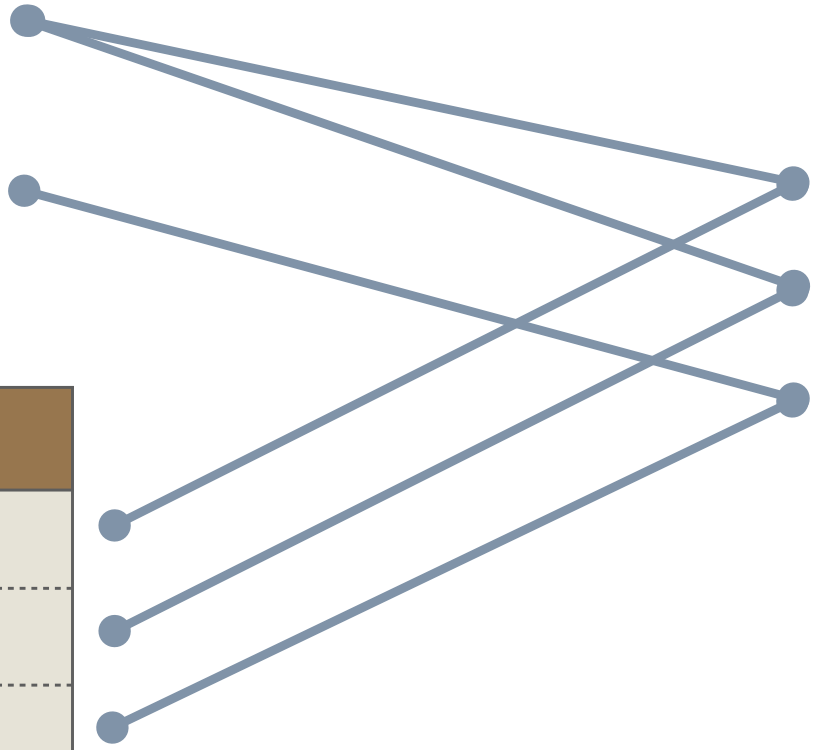
Natural join example

Student

sid	name
1	Jill
2	Bo
3	Maya

Takes

sid	cid
1	445
1	483
3	435



Calculate:

Student ⋈ Takes

sid	name	cid
1	Jill	445
1	Jill	483
3	Maya	435

Theta Join

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

  $=, <, \neq, \leq, >, \geq$

Example: Student $\bowtie_{age>age}$ Prof

Equi-join

- A theta join where θ is an equality
- $R_1 \bowtie_{A=B} R_2 = \Pi_{\Omega}(\sigma_{A=B}(R_1 \times R_2))$
  - Very useful join in practice
  - Keeping both A and B attributes is redundant; projection removes second (here B)

- Example: Student $\bowtie_{sid=sid}$ Takes

i-clicker #5

- Please calculate:

$$\Pi_{\text{name,sid}}(\sigma_{\text{title="DB"}}(\text{Course} \bowtie (\text{Students} \bowtie \text{Takes})))$$

Course

cid	title	sem
445	DB	F08
483	AI	S08
435	Arch	F08

Students

sid	name
1	Jill
2	Bo
3	Maya

Takes

sid	cid
1	445
1	483
3	435

i-clicker #5

$\Pi_{name,sid}(\sigma_{title="DB"}(Course \bowtie (Students \bowtie Takes)))$

- The result table for this query….
  A. contains a record for (only) Jill.
  B. contains records for both Jill and Maya.
  C. contains a record with field "DB"
  D. contains a record with field "F08"
  E. none of the above.

Answer on next slide

i-clicker #5

$\Pi_{name,sid}(\sigma_{title="DB"}(Course \bowtie (Students \bowtie Takes)))$

- The result table for this query….
  - A. contains a record for (only) Jill.
  - B. contains records for both Jill and Maya.
  - C. contains a record with field "DB"
  - D. contains a record with field "F08"
  - E. none of the above.

sid	name
1	Jill

Review

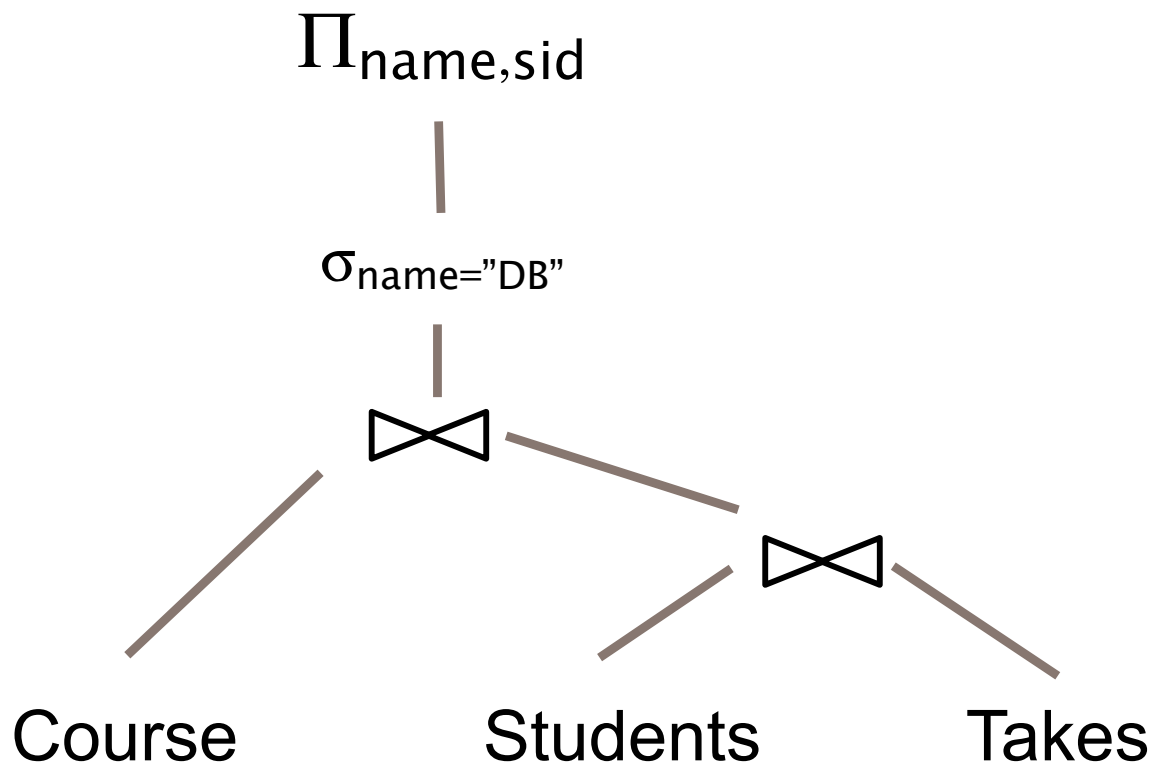
- Five basic operators of the Relational Algebra:
  - Union: $\cup$
  - Difference: -
  - Selection: σ
  - Projection: Π
  - Cartesian Product: $\times$
- Derived or auxiliary operators:
  - Intersection, complement
  - Joins (natural, theta join, equi-join)
  - Renaming: ρ

Natural join questions

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

Combining operators: complex expressions

$\Pi_{name,sid}(\sigma_{name="DB"}(Course \bowtie (Students \bowtie Takes)))$



Query equivalence

Definition: **Query Equivalence**

Two queries Q and Q' are equivalent if:

for all databases D, Q(D) = Q'(D)

Query Optimization Is Based on Algebraic Equivalences

- Relational algebra has laws of commutativity, associativity, etc. that imply certain expressions are **equivalent**.
- They may be different in cost of evaluation!

$$\sigma_{c \wedge d}(R) \equiv \sigma_c(\sigma_d(R)) \qquad \text{cascading selection}$$

$$R \bowtie (S \bowtie T) \equiv (R \bowtie S) \bowtie T \qquad \text{join associativity}$$

$$\sigma_c(R \bowtie S) \equiv \sigma_c(R) \bowtie S \qquad \text{pushing selections}$$

- Query optimization finds the most efficient representation to evaluate (or one that's not bad)

Relational calculus

- What is a “calculus”?
  - The term "calculus" means a system of computation
  - The relational calculus is a system of computing with relations

Relational calculus (in 1 slide)

English: Name and sid of students who are taking the course “DB”

RA: $\Pi_{name,sid}(\text{Students} \bowtie \text{Takes} \bowtie \sigma_{name=\text{"DB"}}(\text{Course}))$

RC: $\{x_{name}, x_{sid} \mid \exists x_{cid} \exists x_{term}\ \text{Students}(x_{sid}, x_{name}) \wedge \text{Takes}(x_{sid}, x_{cid}) \wedge \text{Course}(x_{cid}, \text{"DB"}, x_{term})\}$

Where are the joins?

Algebra v. Calculus

- *Relational Algebra*: More operational; very useful for representing execution plans.
- *Relational Calculus*: More declarative, basis of SQL

- The calculus and algebra have equivalent expressive power (Codd)

A language that can express this core class of queries is called **Relationally Complete**