Relational Model and Relational Algebra

CMPSCI 445 – Database Systems

Fall 2008

Some slide content courtesy of Zack Ives, Ramakrishnan & Gehrke, Dan Suciu, Ullman & Widom

Next lectures: Querying relational data

- Today
  – Relational model, Relational algebra
- Wednesday
  – SQL
- Homework 1 assigned Friday
- Next Week
  – More SQL

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: Definitions

- *Relational database:* a set of *relations*

Relational Database: Definitions

- *Relational database:* a set 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.

Relational Database: Definitions

- *Relational database:* a set 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

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

Relational instances: tables

column, attribute, field

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

Relational instances: tables

column, attribute, field

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

row, tuple

Relational instances: tables

column, attribute, field

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

row, tuple

Attribute *value*

Relational instances: tables

column, attribute, field

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

row, tuple

Attribute *value*

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 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” ?

What is an “Algebra” ?

- Mathematical system consisting of:

What is an “Algebra” ?

- Mathematical system consisting of:
  - *Operands* --- variables or values from which new values can be constructed.

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*
  - Later: we discuss how to extend this to *bags*
- 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

R_1

sid	name
1	Jill
2	Bo
3	Maya

R_2

sid	name
1	Jill
4	Bob

$R_1 \cup R_2$

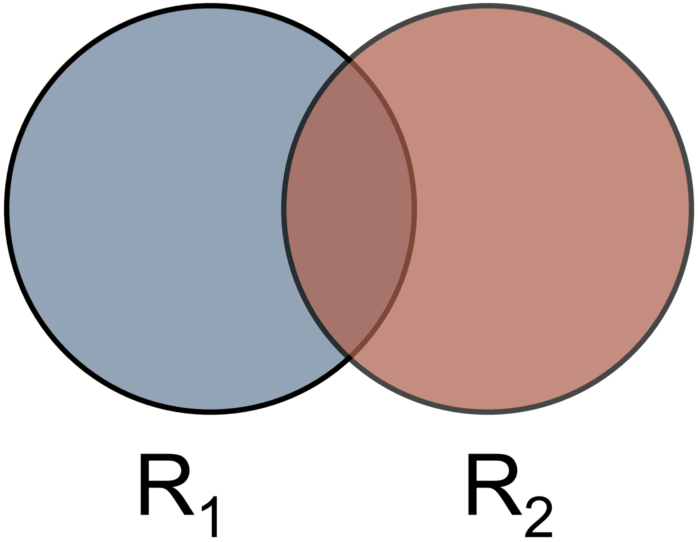
sid	name
1	Jill
2	Bo
3	Maya
4	Bob

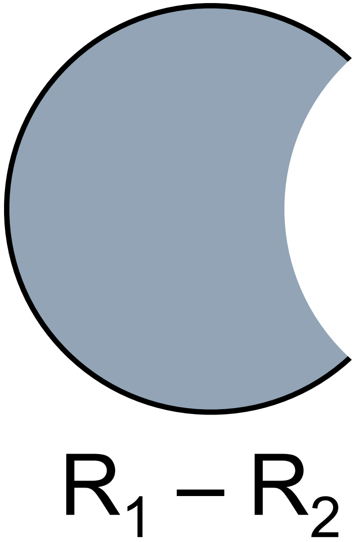
$R_1 - R_2$

sid	name
2	Bo
3	Maya

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)





3. Selection

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

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

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

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

Course

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

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)**

4. Projection

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

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

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

Course

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

$\xrightarrow{\Pi}$

Answer

cid	name
445	DB
483	AI

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

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

Renaming

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

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

Renaming

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

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

Course

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

$\xrightarrow{\rho}$

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

Natural Join

- Notation: $R_1 \bowtie R_2$

- Meaning: $R_1 \bowtie R_2 = \Pi_A(\sigma_C(R_1 \times R_2))$

- Where:
  - The selection σ_C checks equality of all common attributes
  - The projection eliminates 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:

$$\text{Student} \bowtie \text{Takes}$$

Natural join example

Student

sid	name
1	Jill
2	Bo
3	Maya

Takes

sid	cid
1	445
1	483
3	435

Calculate:

Student $\bowtie$ 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: $\text{Student} \bowtie_{\text{age>age}} \text{Prof}$

Equi-join

- A theta join where θ is an equality
- $R_1 \bowtie_{A=B} R_2 = \sigma_{A=B} (R_1 \times R_2)$
- Very useful join in practice

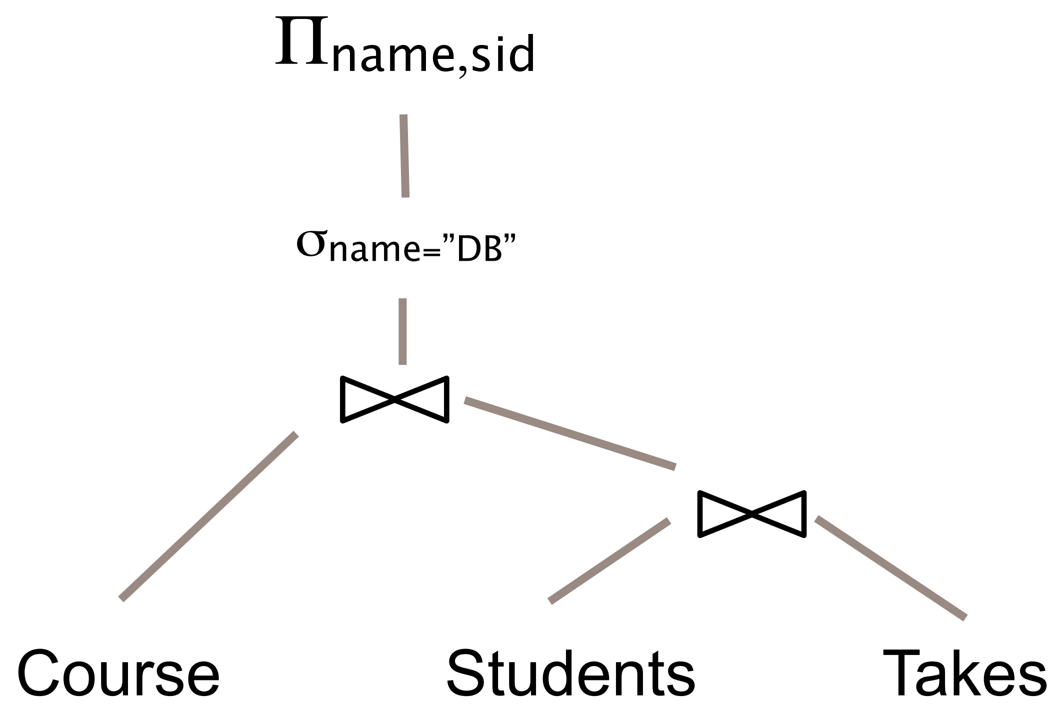
- Example: Student $\bowtie_{sid=sid}$ Takes

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,equi-join, theta join)
  - Renaming: ρ

Combining operators: complex expressions

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



In-class exercise

- Please calculate:

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

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

Query equivalence

Definition: **Query Equivalence**

Two queries Q and Q' are equivalent if:

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

Query equivalence

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!

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

$$R \bowtie (S \bowtie T) \equiv (R \bowtie S) \bowtie T$$ join associativity

$$\sigma_c(R \bowtie S) \equiv \sigma_c(R) \bowtie S$$ 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”

Relational calculus (in 1 slide)

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

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

Relational calculus (in 1 slide)

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

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

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

Relational calculus (in 1 slide)

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

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

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

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**