# RELATIONS AND RELATIONAL ALGEBRA

*Introduction to Database Systems*

*Mahdi Akhi*
*Sharif University of Technology*

# IN THIS LECTURE

- The Relational Model
  - Relational data structure
  - Relational data manipulation

- For more information
  - Connolly and Begg – Chapters 3.1-3.2.2 and 4
  - Ullman and Widom – Chapter 3.1, 5.1

# RELATIONS

➤ We will use tables to represent relations:

| | |
|---|---|
| Ahmad | ahmad@sharif.edu |
| Bahman | bahman@sharif.edu |
| Yasamin | yasamin@sharif.edu |

# RELATIONS

- This is a relation between people and email addresses

| | |
|---|---|
| Ahmad | ahmad@sharif.edu |
| Bahman | bahman@sharif.edu |
| Yasamin | yasamin@sharif.edu |

# RELATIONS

- A mathematician would say that it is a set of pairs:

  <Ahmad, ahmad@sharif.edu>, <bahman@sharif.edu>, and <Yasamin, yasamin@sharif.edu>.

| | |
|---|---|
| Ahmad | ahmad@sharif.edu |
| Bahman | bahman@sharif.edu |
| Yasamin | yasamin@sharif.edu |

# RELATIONS

- Each value in the first column is a *name*, each value in the second column is an *email address*.
- In general, each column has a ***domain*** – a set from which all possible values can come.

| | |
|---|---|
| Ahmad | ahmad@sharif.edu |
| Bahman | bahman@sharif.edu |
| Yasamin | yasamin@sharif.edu |

# RELATIONS

- A mathematical relation is a set of tuples: sequences of values. Each tuple corresponds to a row in a table:

| | | |
|---|---|---|
| Ahmad | ahmad@sharif.edu | 0115111111 |
| Bahman | bahman@sharif.edu | 0115222222 |
| Yasamin | yasamin@sharif.edu | 0115333333 |

# RELATIONS: TERMINOLOGY

- **Degree of a relation:** how long the tuples are, or how many **columns** the table has
    - In the first example (name, email) degree of the relation is 2
    - In the second example (name, email, telephone) degree of the relation is 3
    - Often relations of degree 2 are called binary, relations of degree 3 are called ternary etc.
- ***Cardinality of the relation***: how many different tuples are there, or how many different rows the table has.

# RELATIONS: A QUICK LOOK

Attributes

**Branch**

Relation

| branchNo | street | city | postcode |
|---|---|---|---|
| B005 | 22 Deer Rd | London | SW1 4EH |
| B007 | 16 Argyll St | Aberdeen | AB2 3SU |
| B003 | 163 Main St | Glasgow | G11 9QX |
| B004 | 32 Manse Rd | Bristol | BS99 1NZ |
| B002 | 56 Clover Dr | London | NW10 6EU |

Cardinality

Degree

Primary key

Foreign key

**Staff**

Relation

| staffNo | fName | lName | position | sex | DOB | salary | branchNo |
|---|---|---|---|---|---|---|---|
| SL21 | John | White | Manager | M | 1-Oct-45 | 30000 | B005 |
| SG37 | Ann | Beech | Assistant | F | 10-Nov-60 | 12000 | B003 |
| SG14 | David | Ford | Supervisor | M | 24-Mar-58 | 18000 | B003 |
| SA9 | Mary | Howe | Assistant | F | 19-Feb-70 | 9000 | B007 |
| SG5 | Susan | Brand | Manager | F | 3-Jun-40 | 24000 | B003 |
| SL41 | Julie | Lee | Assistant | F | 13-Jun-65 | 9000 | B005 |

# RELATIONS: MATHEMATICAL DEFINITION

- Mathematical definition of a relation **R** of degree **n**, where values come from **domains $A_1, \ldots, A_n$**:

$$R \subseteq A_1 \times A_2 \times \ldots \times A_n$$

(relation is a subset of the Cartesian product of domains)

**Cartesian product:**

$$A_1 \times A_2 \times \ldots \times A_n =$$

$$\{<a_1, a_2, \ldots, a_n>: a_1 \in A_1, a_2 \in A_2, \ldots, a_n \in A_n\}$$

# RELATIONAL MODEL: DATA MANIPULATION

- Data is represented as **relations**.
- Manipulation of data (query and update operations) corresponds to operations on relations
- Relational algebra describes those operations. They take relations as arguments and produce new relations.
- Think of numbers and corresponding operators +,–,\, * or booleans and corresponding operators &,|,! (and, or, not).
- Relational algebra contains two kinds of operators: **common set-theoretic** ones and operators **specific** to relations (for example projecting on one of the columns).

# UNION

- Standard set-theoretic definition of union:

$$A \cup B = \{x: x \in A \text{ or } x \in B\}$$

- For example, $\{a,b,c\} \cup \{a,d,e\} = \{a,b,c,d,e\}$

- For relations, we want the result to be a relation again: a set $R \subseteq A_1 \times \ldots \times A_n$ for some n and domains $A_1,\ldots,A_n$. (or, in other words, a proper table, with each column associated with a single domain of values).
- So we require in order to take a union of relations R and S that R and S have the **same number of columns** and that corresponding columns have the **same domains**.

# UNION-COMPATIBLE RELATIONS

- Two relations **R** and **S** are **union-compatible** if they have the same number of columns and corresponding columns have the same domains.

# EXAMPLE: NOT UNION-COMPATIBLE

(**different number** of columns)

| | | |
|---|---|---|
| Ahmad | aaa | 111111 |
| Bahman | bbb | 222222 |
| Sina | ccc | 333333 |

| | |
|---|---|
| Tom | 1980 |
| Sam | 1985 |
| Steve | 1986 |

# EXAMPLE: NOT UNION-COMPATIBLE

(**different domains** for the second column)

| | |
|---|---|
| Ahmad | aaa |
| Bahman | bbb |
| Sina | ccc |

| | |
|---|---|
| Tom | 1980 |
| Sam | 1985 |
| Steve | 1986 |

# EXAMPLE: UNION-COMPATIBLE

| | |
|---|---|
| Ahmad | 1970 |
| Bahman | 1971 |
| Sina | 1972 |

| | |
|---|---|
| Roz | 1980 |
| Reza | 1985 |
| Amir | 1986 |

# UNION OF TWO RELATIONS

- Let **R** and **S** be two union-compatible relations. Then their union **R** ∪ **S** is a relation which contains tuples from both relations:

$$R \cup S = \{x: x \in R \text{ or } x \in S\}.$$

- Note that union is a partial operation on relations: it is only defined for some (compatible) relations, not for all of them.

- Similar to division for numbers (result of division by 0 is not defined).

# EXAMPLE: SHOPPING LISTS

**R**

| | |
|---|---|
| Cheese | 1.34 |
| Milk | 0.80 |
| Bread | 0.60 |
| Eggs | 1.20 |
| Soap | 1.00 |

**S**

| | |
|---|---|
| Cream | 5.00 |
| Soap | 1.00 |

**R ∪ S**

| | |
|---|---|
| Cheese | 1.34 |
| Milk | 0.80 |
| Bread | 0.60 |
| Eggs | 1.20 |
| Soap | 1.00 |
| Cream | 5.00 |

# DIFFERENCE OF TWO RELATIONS

Let **R** and **S** be two union-compatible relations. Then their ***difference*** **R – S** is a relation which contains tuples which are in R but not in S:

$$R - S = \{x: x \in R \text{ and } x \notin S\}.$$

- Note that difference is also a partial operation on relations.

# EXAMPLE

**R**

| | |
|---|---|
| Cheese | 1.34 |
| Milk | 0.80 |
| Bread | 0.60 |
| Eggs | 1.20 |
| Soap | 1.00 |

**S**

| | |
|---|---|
| Cream | 5.00 |
| Soap | 1.00 |

**R – S**

| | |
|---|---|
| Cheese | 1.34 |
| Milk | 0.80 |
| Bread | 0.60 |
| Eggs | 1.20 |

# INTERSECTION OF TWO RELATIONS

Let **R** and **S** be two union-compatible relations. Then their ***intersection*** is a relation **R ∩ S** which contains tuples which are both in **R** and **S**:

$$R \cap S = \{x: x \in R \text{ and } x \in S\}$$

- Note that intersection is also a partial operation on relations.

# EXAMPLE

**R**

| | |
|---|---|
| Cheese | 1.34 |
| Milk | 0.80 |
| Bread | 0.60 |
| Eggs | 1.20 |
| Soap | 1.00 |

**S**

| | |
|---|---|
| Cream | 5.00 |
| Soap | 1.00 |

**R ∩ S**

| | |
|---|---|
| Soap | 1.00 |

# CARTESIAN PRODUCT

- Cartesian product is a **total operation** on relations.
- Usual set theoretic definition of product:

$$R \times S = \{<x, y>: x \in R, y \in S\}$$

- Under the standard definition, if $<\text{Cheese}, 1.34> \in R$ and $<\text{Soap}, 1.00> \in S$, then

$$<<\text{Cheese}, 1.34>, <\text{Soap}, 1.00>> \in R \times S$$

(the result is a pair of tuples).

# EXTENDED CARTESIAN PRODUCT

- ***Extended* Cartesian product** flattens the result in a 4-element tuple: <Cheese, 1.34, Soap, 1.00>
- For the rest of the course, "product" means extended product.

# EXTENDED CARTESIAN PRODUCT OF RELATIONS

Let **R** be a relation with column domains $\{A_1,\ldots,A_n\}$ and **S** a relation with column domains $\{B_1,\ldots,B_m\}$. Then their **Extended Cartesian Product** $R \times S$ is a relation

$$R \times S = \{<c_1,\ldots,c_n,c_{n+1},\ldots,c_{n+m}>: \\ <c_1, \ldots,c_n> \in R,\ <c_{n+1},\ldots,c_{n+m}> \in S\}$$

# EXAMPLE

R

| | |
|---|---|
| Cheese | 1.34 |
| Milk | 0.80 |
| Bread | 0.60 |
| Eggs | 1.20 |
| Soap | 1.00 |

S

| | |
|---|---|
| Cream | 5.00 |
| Soap | 1.00 |

R×S

| | | | |
|---|---|---|---|
| Cheese | 1.34 | Cream | 5.00 |
| Milk | 0.80 | Cream | 5.00 |
| Bread | 0.60 | Cream | 5.00 |
| Eggs | 1.20 | Cream | 5.00 |
| Soap | 1.00 | Cream | 5.00 |
| Cheese | 1.34 | Soap | 1.00 |
| Milk | 0.80 | Soap | 1.00 |
| Bread | 0.60 | Soap | 1.00 |
| Eggs | 1.20 | Soap | 1.00 |
| Soap | 1.00 | Soap | 1.00 |

# PROJECTION

- Let **R** be a relation with **n** columns, and **X** is a set of column identifiers (at the moment, we will use numbers, but later we will give them names, like "Email", or "Telephone"). Then projection of **R** on **X** is a new relation $\pi_X(R)$ which only has columns from **X**.

- For example, $\pi_{1,2}(R)$ is a table with only the **1**st and **2**nd columns from **R**.

## EXAMPLE: $\Pi_{13}$ ( R)

R:

| 1 | 2 | 3 |
|---|---|---|
| Ahmad | aaa@ilam.ac.ir | 0115111111 |
| Bahman | bbb@ilam.ac.ir | 0115222222 |
| Sina | sss@ilam.ac.ir | 0115333333 |

# EXAMPLE: $\Pi_{13}$ ( R)

$\pi_{13}$ ( R):

| | |
|---|---|
| Ahmad | 0115111111 |
| Bahman | 0115222222 |
| Sina | 0115333333 |

# SELECTION

- Let **R** be a relation with **n** columns and $\alpha$ is a property of tuples.

- **Selection from R subject to condition α** is defined as follows:

$$\sigma_{\alpha}(R) = \{<a_1, \ldots, a_n> \in R : \alpha(a_1, \ldots, a_n)\}$$

# WHAT IS A REASONABLE PROPERTY?

- We assume that properties are written using {and, or, not} and expressions of the form **col(i)** $\Theta$ **col(j)** (where i,j are column numbers) or **col(i)** $\Theta$ **v**, where **v** is a value from the domain $\mathbf{A}_i$.

- $\Theta$ is a comparator which makes sense when applied to values from i and j columns ($=$ , $\neq$, maybe also $\leq$, $\geq$, $<$, $>$ if there is a natural order on values).

# WHAT IS A MEANINGFUL COMPARISON

- We can always at least tell if two values in the same domain are **equal or not** (database values are finitely represented).
- In some cases, it makes sense to **compare** values from different column domains: for example, if both are domains contain strings, or both contain dates.
- For example, 1975 > 1987 is a meaningful comparison, "Ahmad" = 1981 is not.
- We can only use a comparison in selection property if its result is true or false, never undefined.

# EXAMPLE: SELECTION

- $\sigma_{\text{col}(3) < 2002 \text{ and col}(2) = \text{Nolan}}(R)$

R

| | | |
|---|---|---|
| Insomnia | Nolan | 2002 |
| Magnolia | Anderson | 1999 |
| Insomnia | Sobasa | 1997 |
| Memento | Nolan | 2000 |
| Gattaca | Niccol | 1997 |

# EXAMPLE: SELECTION

- $\sigma_{\text{col}(3) < 2002 \text{ and col}(2) = \text{Nolan}}(R)$

| | | |
|---|---|---|
| Insomnia | Nolan | 2002 |
| Magnolia | Anderson | 1999 |
| Insomnia | Sobasa | 1997 |
| Memento | Nolan | 2000 |
| Gattaca | Niccol | 1997 |

# EXAMPLE: SELECTION

- $\sigma_{\text{col}(3) < 2002 \text{ and col}(2) = \text{Nolan}} (R)$

| | | |
|---|---|---|
| Insomnia | Nolan | 2002 |
| Magnolia | Anderson | 1999 |
| Insomnia | Sobasa | 1997 |
| Memento | Nolan | 2000 |
| Gattaca | Niccol | 1997 |

# EXAMPLE: SELECTION

- $\sigma_{\text{col}(3) < 2002 \text{ and col}(2) = \text{Nolan}}(R)$

| | | |
|---|---|---|
| Insomnia | Nolan | 2002 |
| Magnolia | Anderson | 1999 |
| Insomnia | Sobasa | 1997 |
| Memento | Nolan | 2000 |
| Gattaca | Niccol | 1997 |

# EXAMPLE: SELECTION

- $\sigma_{\text{col(3)} < 2002 \text{ and col(2)} = \text{Nolan}} (R)$

| | | |
|---|---|---|
| Insomnia | Nolan | 2002 |
| Magnolia | Anderson | 1999 |
| Insomnia | Sobasa | 1997 |
| Memento | Nolan | 2000 |
| Gattaca | Niccol | 1997 |

# EXAMPLE: SELECTION

- $\sigma_{\text{col}(3) < 2002 \text{ and col}(2) = \text{Nolan}}(R)$

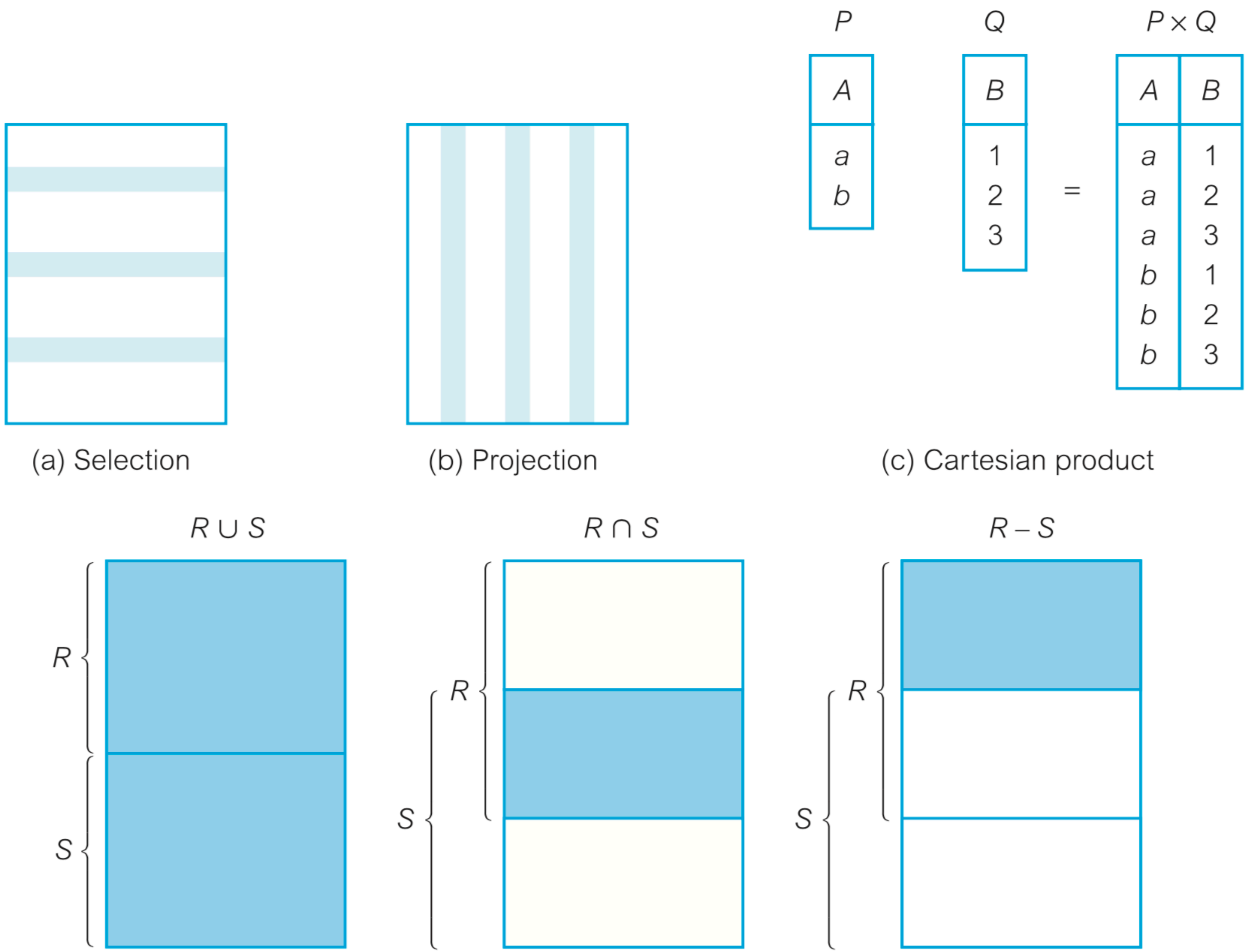
| | | |
|---|---|---|
| Insomnia | Nolan | 2002 |
| Magnolia | Anderson | 1999 |
| Insomnia | Sobasa | 1997 |
| Memento | Nolan | 2000 |
| Gattaca | Niccol | 1997 |

# EXAMPLE: SELECTION

- $\sigma_{\text{col}(3) < 2002 \text{ and col}(2) = \text{Nolan}}(R)$

| | | |
|---|---|---|
| Memento | Nolan | 2000 |

# SUMMARY SO FAR!




(d) Union (e) Intersection (f) Set difference

# SUMMARY SO FAR

- Data is represented as tables
- Operations on tables:
  - **Union** of two union-compatible tables (tables with the same number of columns and the same domains for corresponding columns)
  - Set **difference** of two union-compatible tables
  - **Intersection** of two union-compatible tables
  - **Extended Cartesian** product of two tables
  - **Project** a table on some of its columns
  - **Select** rows in a table satisfying a property
- Result of an operation is again a table, so operations can be chained

# THETA JOIN (Θ-JOIN)

$$\mathbf{R} \bowtie_F \mathbf{S}$$

- Contains tuples satisfying the predicate F from the Cartesian product of **R** and **S**

$$R \bowtie_F S = \sigma_F(R \times S)$$

# NATURAL JOIN

## R ⋈ S

- Joint of the two relations **R** and **S** over all common attributes **x**.
- One occurrence of each common attribute is eliminated from the result.

# LEFT-OUTER JOIN

$$\mathbf{R} ⟕ \mathbf{S}$$

- Outer join is a join in which tuples from **R** that do not have matching values in the common attributes of **S** are also included in the result relation.
- Missing values in the second relation are set to null.

## SEMI-JOIN

$$\mathbf{R} \triangleright_F \mathbf{S}$$

- The Semi-join operation defines a relation that contains the tuples of *r* that participate in the join of **R** with **S** satisfying the predicate *F*

$$R \triangleright_F S = \Pi_A(R \bowtie_F S)$$

# JOIN EXAMPLES

$T$

| $A$ | $B$ |
|---|---|
| $a$ | 1 |
| $b$ | 2 |

$U$

| $B$ | $C$ |
|---|---|
| 1 | $x$ |
| 1 | $y$ |
| 3 | $z$ |

$T \bowtie U$

| $A$ | $B$ | $C$ |
|---|---|---|
| $a$ | 1 | $x$ |
| $a$ | 1 | $y$ |

(g) Natural join

$T \rhd_B U$

| $A$ | $B$ |
|---|---|
| $a$ | 1 |

(h) Semijoin

$T ⟕_B U$

| $A$ | $B$ | $C$ |
|---|---|---|
| $a$ | 1 | $x$ |
| $a$ | 1 | $y$ |
| $b$ | 2 | |

(i) Left Outer join

# OTHER OPERATIONS

- Not all SQL queries can be translated into relational algebra operations defined in the lecture.
- Extended relational algebra includes counting and other additional operations.

# EXAMPLE EXAM QUESTION

- What is the result of

$\pi_{1,3}(\sigma_{\text{col}(2) = \text{col}(4)} (R \times S) )$, where R and S are:

R

| | |
|---|---|
| Ahmad | 111111 |
| Bahman | 222222 |

S

| | |
|---|---|
| Sina | 333333 |
| Danial | 111111 |

(5 marks)

# END

*Thanks to Mohammad Tanhaei, Assistant Prof. At Ilam University*