RELATION AND RELATIONAL OPERATIONS

Relation:

- A table with a distinct name for each column (*attribute*).
- Each attribute A_i has associated with it a *domain* D_i of possible values that may appear in that column.
- Each row of the table is a *tuple* of attribute values, one per column.[†]

Example. STUDENT-GRADES relation.

S#	Sname	C#	Grade
12	John	c1	A
12	John	c2	B
12	John	c3	A
15	Bill	c1	A
15	Bill	c2	A
27	Linda	c1	A
27	Linda	c2	B
31	Betty	c1	A

- There is no ordering of the columns or the rows, i.e., a change in either (or both) does not change the relation.

† Unlike a vector, the tuple items can be of different types.

OPERATIONS ON A RELATION

Two Dimensions of Relations: Horizontal and vertical.

- The schema gives the columns (horizontal dimension) or the *intension*.

STUDENT-GRADES(S#, Sname, C#, Grade).

- The rows (vertical dimension) form the *extension* or *value* of the relation schema.
- The extension may change over time as rows are added or deleted or updated. The schema of a relation remains fixed.

Operations: Two types for the two dimensions.

- Operations that modify the extension (vertical dimension).

Adds rows:	Union
Subtracts rows:	Intersection, Difference, Subset formation (selection)

- Operations that modify intention (and maybe extension also):

Add columns:	Cartesian product, Join
Subtract columns:	Projection, Division

SET THEORETIC OPERATIONS: TYPE-I

- The two operands (relation instance) in each of union, intersection, and difference operation must have the *same* schema.

Example. Consider the schema STUDENT(S#, Sname).

extension R_1

S#	Sname
12	John
27	Linda
31	Betty
44	Steve

extension R_2

S#	Sname
12	John
27	Linda
32	Linda

Union:

- $R_1 \cup R_2$ = the set of all tuples which belong to one or both of R_1 and R_2; $|R_1 \cup R_2| \leq |R_1| + |R_2|$.

$R_1 \cup R_2$

S#	Sname
12	John
27	Linda
31	Betty
44	Steve
32	Linda

extension R_1

S#	Sname
12	John
27	Linda
31	Betty
44	Steve

extension R_2

S#	Sname
12	John
27	Linda
32	Linda

Intersection:

- $R_1 \cap R_2$ = the set tuples in both R_1 and R_2; $|R_1 \cap R_2| \leq \min(|R_1|, |R_2|)$.

S#	Sname
12	John
27	Linda

Difference:

- $R_1 - R_2$ = the set of tuples in R_1 but not in R_2; $|R_1 - R_2| \leq |R_1|$.

S#	Sname
31	Betty
44	Steve

Selection:

- $\sigma_P(R_1)$ = the set of tuples in R_1 which satisfy the predicate P; $|\sigma_P(R_1)| \leq |R_1|$. Let P = "even $S\#$".

S#	Sname
12	John
44	Steve

OTHER OPERATIONS: TYPE II

Cartesian product:

- $R_1 \times R_2$ = all combinations of tuples of R_1 and tuples of R_2; $|R_1 \times R_2| = |R_1| \times |R_2|$.

STUDENT

S#	Sname
12	John
27	Linda
31	Betty

COURSE

C#	Cname
c1	Database
c2	Compiler
c3	Pascal
c4	Cobol

STUDENT $\times$ COURSE

S#	Sname	C#	Cname
12	John	c1	Database
12	John	c2	Compiler
12	John	c3	Pascal
12	John	c4	Cobol
27	Linda	c1	Database
27	Linda	c2	Compiler
27	Linda	c3	Pascal
27	Linda	c4	Cobol
	...		...

- What is R_1(A, B, C)$\times R_2$(A, D), where attribute A is common?

  The result relation has attrbuites ($A^{(1)}$, B, C, $A^{(2)}$, D), where $A^{(1)}$ and A^2 captures attribute A in R_1 and A^2, respectively. (Note that A in R^1 may have a different meaning than A in R_2.)

Projection:

- $\Pi_A(R)$ = from each tuple in R take only the values in the set of columns in A (eliminate any duplicate rows that may be generated in the process); $|\Pi_A(R)| \leq |R|$.
- Projection can be taken on a set of attributes.

STUDENT

S#	Sname
12	John
15	Bill
27	Linda
32	Linda

Π_{Sname}(STUDENT)

Sname
John
Bill
Linda

$\Pi_{S\#}$(STUDENT)

S#
12
15
27
32

Natural Join:

- $R_1 \otimes R_2$ = the tuples of $R_1 \times R_2$ for which the values in the common columns of R_1 and R_2 are identical; $|R_1 \otimes R_2| \leq |R_1| \times |R_2|$.
- Only one set of the common columns are kept.

STUDENT

S#	Sname
12	John
27	Linda
31	Betty
44	Steve

GRADE

S#	C#	Grade
12	c1	A
12	c2	B
27	c1	B
31	c2	A

STUDENT $\otimes_{S\#}$ GRADE

S#	Sname	C#	Grade
12	John	c1	A
12	John	c2	B
27	Linda	c1	B
31	Betty	c2	A

- The join-relation shows all students and their course grades.
- Since Steve has no grades (has not taken any courses), his information is lost.
- The join STUDENT $\otimes_{S\#}$ GRADE can be expressed in terms of selection and projection as

$$\Pi_{(S\#, Sname, C\#, Grade)}\left[\sigma_{STUDENT.S\#=GRADE.S\#}(STUDENT \times GRADE)\right]$$

Division:

- R_1/R_2, where Attrb(R_2) $\subset$ Attrb(R_1). The result relation has attributes Attrb(R_1) – Attrb(R_2).
  - (1) First obtain the projection of R_1 on the columns other than those in R_2.
  - (2) Then, select those rows of the projection whose cartesian product with R_2 is contained in R_1.

R_1 = STUDENT_GRADE

S#	Sname	C#	Grade
12	John	c1	A
12	John	c2	B
12	John	c3	A
15	Bill	c1	B
15	Bill	c2	B
15	Bill	c3	A
27	Linda	c1	A
27	Linda	c2	B
31	Betty	c1	A

R_2 = GRADE

C#	Grade
c1	A
c2	B

R_1/R_2

S#	Sname
12	John
27	Linda

- R_1/R_2 gives (S#, Sname) of those students who received grade A in course c1 and grade B in course c2.
- No single row in R_1 = STUDENT_GRADE can give the information about both (c1, A) and (c2, B) in GRADE.
- The division operator is used when several rows have to be evaluated together with separate criteria applied to each of those rows. We can express R_1/R_2 above using projection and selections operations as follows

$$\Pi_{(S\#,Sname)}\sigma_{(C\#=c1)\wedge(Grade=A)}(R_1) \cap \Pi_{(S\#,Sname)}\sigma_{(C\#=c2)\wedge(Grade=B)}(R_1)$$

EXERCISE

1. How can you obtain (S#, Sname) of STUDENTs who satisfy the criteria P: "(A in c1) or (B in c2)"? For the relation given on the previous page, you should get the following final answer.

S#	Sname
12	John
15	Bill
27	Linda
31	Betty

2. Find the expression involving division and other operations to determine (S#, Sname) of students who had taken at least the courses c1 and c2 and received at least one A (may be in a course different c1 and c2). Keep the number of relational-operations performed as small as possible. (Note that finding (S#, Sname) for students who received at least two *A*'s is more complex.)
3. Which of $(R_1/R_2)\times R_2 \subseteq R_1$ and $(R_1/R_2)\times R_2 \supseteq R_1$ is true for abitrary R_1 and R_2 with Attrb(R_2) $\subset$ Attrb(R_1)?
4. Is it better to have a relation COURSE-GRADE(S#, C#, Grade) for all courses or have several relations like DATABASE-GRADE(S#, Grade), THEORY-GRADE(S#, Grade), etc, one relation for each course? Which form requires less total storage to represent a given collection of course-grade information?

DECOMPOSITION OF A RELATION

- For disjoint sets of attributes X, Y, and Z, a relation R(X,Y,Z) is *decomposable* if R = Π_{XY}R $\otimes_{Y}$ Π_{YZ}R.

Example:

- For the example R(A, B, C) below R $\neq$ Π_{AB}R $\otimes_{B}$ Π_{BC}R. The tuple (a1, b1, c2) is in the join but not in R.

A	B	C		A	B		B	C
a1	b1	c1		a1	b1		b1	c1
a1	b2	c2	$\neq$	a1	b2	$\otimes$	b2	c2
a2	b3	c1		a2	b3		b3	c1
a2	b1	c2		a2	b1		b1	c2

- R $\neq$ Π_{XY}R $\otimes_{Y}$ Π_{YZ}R means the facts described by (X,Y,Z) do not consists of independent facts described by (X,Y) and (Y,Z).
- If A = Person, B = Project, and C = Hours-worked, then the decomposition holds if each person in a project works the same number of hours. That is, c1 = c2 = the hours for project b1.

A	B	C		A	B		B	C
a1	b1	c1		a1	b1		b1	c1
a1	b2	c2	=	a1	b2	$\otimes$	b2	c2
a2	b3	c1		a2	b3		b3	c1
a2	b1	c1		a2	b1			

DECOMPOSITION THEOREM

Theorem.

- If Y $\to$ Z is a functional dependency, then R(X, Y, Z) = Π_{XY}R $\otimes_Y$ Π_{YZ}R.

EXERCISE

1. If we impose B $\to$ A instead, what changes we need to make in the fourth tuple in R(A,B,C). Show $R(A, B)$ and $R(B, C)$ after making the change and verify $R(A, B, C) = R(A, B) \otimes R(B, C)$.