Module 5: Relations

Reading from the Textbook: Chapter 3 Functions, Sequences, and Relations

Introduction

As we saw in the last module, a function is a many-to-one mapping. Relations generalize the concept to many-to-many mappings. Examples of familiar relations are:

- Parent-child relation (where each parent may have several children and each child has two parents),
- Sibling relation (where each person may have several siblings), and
- Algebraic relations like $a \leq b$, where each value of *a* is $\leq$ many values of *b*, and many values of *a* is $\leq$ each value of *b*.

In this module, we study

1. Definitions and Properties of relations,
2. Matrices as applied to relations, and
3. Relational Databases.

Contents

Relation Definitions

A relation is a many-to-many mapping from a set X to a set Y, where

- Each element in *X* may be *related* to many elements in *Y,* and
- Many elements in *X* may be *related* to each element in *Y*.

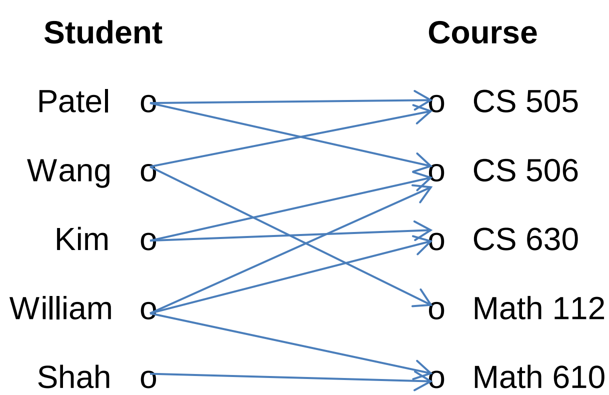
For example, a relation Student-Course, which gives information about students' registration, consists of ordered pairs (s, c) where student *s* is taking course *c*. The relation may be shown as a table where each row includes one ordered pair in the relation. For example, the following table shows the Student-Course relation. The table shows Patel is taking {CS 505, CS 506}, Wang is taking {CS 505, Math 112}, and so on.

Table of Student-Course Relation

Student	Course
Patel	CS 505
Patel	CS 506
Wang	CS 505
Wang	Math 112
Kim	CS 506
Kim	CS 630
William	Math 610
William	CS 506
William	CS 630
Shah	Math 610

A relation *R* from *X* to *Y* may also be shown in the form of a graph, with the left column of vertices corresponding to set *X* and the right column of vertices corresponding to set *Y*. For each ordered pair $(x, y) \in R$, we draw an arrow from vertex *x* to vertex *y*. The graph of Student-Course Relation is shown below. Observe that in general there are several arrows coming out of each vertex on the left, and several arrows comings into each vertex on the right, clearly showing a many-to-many mapping.

Graph of Student-Course Relation



Below is a formal definition of a relation in terms of a set of ordered pairs.

Definition: A binary relation *R* from a set *X* to a set *Y* is a subset of the Cartesian product $X \times Y$. If $(x, y) \in R$, we say *x* is related to *y*, and we write *x R y*.
The sets *X* and *Y* may be the same, $X = Y$. In this case, we say the relation *R* is on the set *X*.

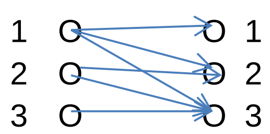
Example: The relation ≤ on the set of integers {1,2,3} is the set of ordered pairs (x, y), where $x \leq y$. So,

$$R = \{(1,1), (1,2), (1,3), (2,2), (2,3), (3,3)\}$$

The table of this relation is shown below.

1	1
1	2
1	3
2	2
2	3
3	3

And the graph of this relation is shown below.



Properties of Relations

We now define several properties that relations may satisfy, namely

- reflexive
- symmetric
- antisymmetric
- transitive

These properties are defined for a relation *R* on a set *X* (that is, from set *X* to set *X*).

Definition (Reflexive): A relation *R* on a set *X* is *reflexive* if and only if

$$\forall x \in X, \qquad (x,x) \in R.$$

Therefore, a relation is not reflexive if

$$\exists x \in X, \qquad (x,x) \notin R.$$

Example: The relation $\leq$ on the set of integers is reflexive, because for every integer *x*, clearly $x \leq x$.

Example: The following relation *R* on the set of integers {1,2,3,4} is not reflexive because $(2,2) \notin R$ and $(4,4) \notin R$.

$$R = \{(1,1),(1,3),(2,3),(2,4),(3,3)\}$$

Definition (Symmetric): A relation *R* on a set *X* is *symmetric* if and only if

$$\forall x \in X \text{ and } \forall y \in X, \qquad \text{if } (x,y) \in R \text{ then } (y,x) \in R.$$

Therefore, a relation is not symmetric if there exists at least one pair *x* and *y* in *X*, such that $(x,y) \in R$ and $(y,x) \notin R$.

Example: Consider a relation R_1 on the set of integers {1,2,3,4} defined by the rule:

$$R_1 = \{(x,y) \mid x \textit{ and } y \textit{ are either both even or both odd}\}$$

In terms of the set of ordered pairs, the relation is

$$R_1 = \{(1,1),(1,3),(2,2),(2,4),(3,1),(3,3),(4,2),(4,4)\}.$$

Clearly this relation is symmetric because for every $(x,y) \in R_1$, there is also $(y,x) \in R_1$.

Example: The following relation, R_2, is not symmetric because $(1,2) \in R$, but $(2,1) \notin R$.

$$R_2 = \{(1,1), (1,2), (1,3), (2,2), (2,3), (3,1), (3,2), (3,3)\}$$

Example: The relation $\leq$ on the set of integers is not symmetric because, for example, $5 \leq 6$ but $\neg(6 \leq 5)$.

Definition (Antisymmetric): A relation *R* on a set *X* is *antisymmetric* if and only if
$$\forall x \in X \text{ and } \forall y \in X, \text{where } (x \neq y), \quad \text{if } (x, y) \in R \text{ then } (y, x) \notin R.$$

Therefore, a relation is not antisymmetric if there exists at least one pair *x* and *y* where $x \neq y$, and both $(x, y) \in R$ and $(y, x) \in R$.

Note that if a relation is symmetric, then it will not be antisymmetric. But if a relation is not symmetric, then it may or may not be antisymmetric.

Example: The relation $\leq$ on the set of integers is antisymmetric because for every pair of integers *a* and *b*, where $a \neq b$, $if\ a \leq b\ then\ \neg(b \leq a)$.

Example: Let's look at the above relation, R_2. (We saw the relation is not symmetric.)

$$R_2 = \{(1,1), (1,2), (1,3), (2,2), (2,3), (3,1), (3,2), (3,3)\}$$

The relation is also not antisymmetric because $(1,3) \in R$ and $(3,1) \in R$.

Next, we discuss the transitive property. First, as an example of transitive property, we all know from regular algebra that if $a \leq b$ and $b \leq c$ then clearly $a \leq c$.

Definition (Transitive): A relation *R* on a set *X* is *transitive* if and only if

$$\forall x \in X, \forall y \in X, \forall z \in X, \quad \text{if } (x, y) \in R \text{ and } (y, z) \in R, \quad \text{then } (x, z) \in R.$$

Therefore, a relation is not transitive if

$$\exists x \exists y \exists z, \quad (x, y) \in R \wedge (y, z) \in R \wedge (x, z) \notin R.$$

Note: $x, y,$ and z do not need to be distinct. For example, let $x = z$. Then the transitive property becomes

$$\forall x \in X, \forall y \in X, \quad \text{if } (x,y) \in R \text{ and } (y,x) \in R, \quad \text{then } (x,x) \in R.$$

This means that if a relation is both transitive and symmetric, then it must be reflexive!

Example: Consider the relation $\leq$ on the set of integers. Formally, this relation is defined as

$$R = \{(x,y) \mid x \leq y\}$$

This relation is clearly transitive because for all integers $x, y, z,$

$$\text{if } x \leq y \text{ and } y \leq z \text{ then } x \leq z.$$

Example: The following relation is not transitive because $(2,3) \in R$ and $(3,1) \in R$, but $(2,1) \notin R$.

$$R_2 = \{(1,1), (1,2), (1,3), (2,2), (2,3), (3,1), (3,2), (3,3)\}$$

The composition of two relations is defined similar to composition of functions.

Definition: Given a relation R_1 from X to Y and a relation R_2 from Y to Z, the composition $R_2 \text{o } R_1$ is a relation from X to Z, defined as

$$R_2 \text{o } R_1 = \{(x,z) | \exists y \in Y, (x,y) \in R_1 \text{ and } (y,z) \in R_2\}$$

Note that the relation $R_1$is applied first, then the relation R_2.

Example: Let $R_1 = \{(1,a), (1,b), (2,a), (2,b), (3,c)\}$ and $R_2 = \{(a,x), (a,z), (b,y), (c,z)\}$.

Then,

$$R_2 \text{o } R_1 = \{(1,x), (1,y), (1,z), (2,x), (2,y), (2,z), (3,c)\}.$$

Next, we discuss relations that group the above properties in some interesting ways.

Partial Order

Consider a relation on the set of NJIT courses that defines prerequisite ordering of courses.

$$R = \{(x, y) | \ x \text{ is a prerequisite of } y, \text{or } x = y\}$$

This relation is

- Reflexive: since $x = y$ is included in the definition.
- Antisymmetric: If x is a prerequisite of y, then y cannot be a prerequisite of x.
- Transitive: If x is a prerequisite of y and y is a prerequisite of z, then x is an indirect prerequisite of z.

Not every pair of courses is ordered by this relation. There are clearly pairs of courses *x* and *y* that may be taken in any order. This relation is called a *partial order*.

As another example, consider the relation ≤ on the set of integers. This relation is reflexive, antisymmetric, and transitive. In addition, every pair of integers x and y are ordered so that either $x \leq y$ or $y \leq x$. This relation is called a *total order*.

Definition (Partial Order): A relation *R* on a set *X* is called a *partial order* if it is

- reflexive
- antisymmetric
- transitive

The reason for naming the relation a *partial* order is that not every pair of elements in *X* is necessarily related. That is, there may be pairs of *x* and *y* in *X* such that $(x, y) \notin R$ and $(y, x) \notin R$.

The relation is called a *total order* if in addition, every pair of elements in X are related so that either $(x, y) \in R$ or $(y, x) \in R$.
(A total order is a special case of partial order, with an additional condition.)

Example: Consider the following relation on the set of positive integers

$$R = \{(x, y) | x \text{ divides } y\}$$

For example, $(2,18) \in R$ because 2 divides 18. (2 is a factor of 18.) This relation is reflexive, antisymmetric, and transitive. So it is a partial order. Furthermore, there are pairs of integers that are not related. For example, integers 2 and 15 are not related because neither one is a factor of the other. So, the relation is not a total order.

Equivalence Relations

The relation = on the set of integers is a trivial example of a relation that is reflexive, symmetric, and transitive. Relations that satisfy these three properties are called equivalence relations. (They behave similar to the equality relation.)

Example: Consider the relation on the set $X = \{0,1,2,\cdots,15\}$ defined as follows.

$$R = \{(x, y) \mid (x \bmod 4) = (y \bmod 4)\}$$

The above condition means that *x* and *y* in binary are equal in their rightmost two bits.

We may explicitly list the ordered pairs in this elation.

$$
\begin{array}{llll}
R = \{ & & & \\
(0,0), & (0,4), & (0,8), & (0,12), \\
(1,1), & (1,5), & (1,9), & (1,13), \\
(2,2), & (2,6), & (2,10), & (2,14), \\
(3,3), & (3,7), & (3,11), & (3,15), \\
\cdots\} & & &
\end{array}
$$

This relation partitions the set of 16 numbers into four disjoint blocks (disjoint subsets):

Block 0: {0, 4, 8, 12}	Rightmost two binary bits of these numbers is	00
Block 1: {1, 5, 9, 13}	Rightmost two binary bits of these numbers is	01
Block 2: {2, 6, 10, 14}	Rightmost two binary bits of these numbers is	10
Block 3: {3, 7, 11, 15}	Rightmost two binary bits of these numbers is	11

Each block is called an *equivalence class*. Two elements *x* and *y* are in the same block (same equivalence class) if and only if $(x, y) \in R$.

■

Definition: A relation *R* on a set *X* is called an *equivalence* relation if the relation is

- Reflexive
- Symmetric
- Transitive

An equivalence relation on *X* partitions elements of *X* into disjoint blocks, where two elements *x* and *y* are in the same block if and only if $(x, y) \in R$.
Each block is called an *equivalence class*.

Matrices of Relations

We now discuss how to represent a relation by a matrix and how to use the matrix to test for various properties of the relation.

Suppose *R* is a relation on the set of integers $\{1,2,3,\cdots,n\}$. Then we can represent the relation by an $n \times n$ Boolean matrix *A*, where

$$A[i,j] = \begin{cases} 1, & if\ (i,j) \in R \\ 0, & if\ (i,j) \notin R. \end{cases}$$

For example, consider the following relation on integers $\{1,2,3\}$:

$$R = \{(1,1), (1,3), (2,1), (2,2), (2,3), (3,2), (3,3)\}$$

This relation is represented by a 3×3 matrix:

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

Testing Various Properties

It is easy to see how we test various properties of a relation *R* by examining its matrix *A*. The earlier definition of each property directly translates in terms of the matrix.

Reflexive: A relation (with matrix *A*) is reflexive if and only all diagonal entries of the matrix are 1. That is,

$$\forall i, \qquad A[i,i] = 1.$$

Symmetric: A relation (with matrix *A*) is symmetric if and only if

$$\forall i \forall j, \qquad A[i,j] = A[j,i]$$

(Each pair of entries symmetrical across the diagonal must be equal.)

Antisymmetric: A relation (with matrix *A*) is antisymmetric if and only if

$$\forall i \forall j, \text{where } i \neq j, \qquad \text{if } A[i,j] = 1 \text{ then } A[j,i] = 0.$$

(This means that $A[i,j]$ and $A[j,i]$ cannot be both 1, but they may be both 0.)

Note this condition is not the same as saying $A[i,j] \neq A[j,i]$. (They may be both 0.)

Example: For the above relation R with the 3×3 matrix, it is easy to verify it is reflexive; it is not symmetric (because $A[1,2] = 0$ and $A[2,1] = 1$); and it is also not antisymmetric (because $A[2,3] = 1$ and $A[3,2] = 1$).

Transitive: A relation (with Boolean matrix *A*) is transitive if and only if

$$T_1: \quad \forall i \forall j \forall k,\ \text{if } (A[i,k]=1 \wedge A[k,j]=1) \text{ then } A[i,j]=1.$$

An equivalent statement is

$$T_2: \quad \forall i \forall j \ \text{ if } \ \exists k, (A[i,k]=1 \wedge A[k,j]=1) \quad \text{then } A[i,j]=1.$$

To see that T_1 and T_2 are equivalent, observe the negation of both becomes:

$$\neg T: \qquad \exists i \exists j\, \exists k,\ (A[i,k]=1 \wedge A[k,j]=1) \ \wedge \ A[i,j]=0$$

Example: Suppose a relation has the matrix:

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

- It is not reflexive, because $A[2,2]=0$.
- It is not symmetric, because $A[1,3] \neq A[3,1]$.
- It is not antisymmetric, because $A[1,2]=1 \ and \ A[2,1]=1$.
- It is not transitive, because $(A[2,1]=1 \wedge A[1,2]=1) \ but \ A[2,2] \neq 1$.

A Simple Program to Test Transitivity

The following program tests if a relation is transitive by enumerating all possible values for i, j, k. The program returns true if the relation is transitive. The program runs in $O(n^3)$ time, which informally means in time proportional to n^3. (The O( ) is read as "order of" and will be formally defined later.)

```
for i = 1 to n
  for j = 1 to n
     for k = 1 to n
         if (A[i,k] = 1 ∧ A[k,j] = 1 ∧ A[i,j] = 0)
             return FALSE
return TRUE
```

Next, we discuss how to test transitivity by matrix multiplication, which offers a more efficient approach than the above program.

Matrix Multiplication: A Review

Let *A* and *B* be two $n \times n$ matrices. Their product is an $n \times n$ matrix $C = A \times B$, where

$$C[i,j] = \sum_{k=1}^{n} A[i,k] * B[k,j]$$

That is, to compute each product term $C[i,j]$, we take row *i* of matrix *A*, column *j* of matrix *B*, and compute their inner product.

If we use this formula in a straightforward way to compute the product, each term $C[i,j]$ takes $O(n)$ time, so the entire matrix *C* takes $O(n^3)$ time. However, there are more advanced algorithms for matrix multiplication that run more efficiently. For example, Strassen's algorithm, which uses a divide-and-conquer strategy, runs in time $O\left(n^{\log_2 7}\right) \cong O(n^{2.8})$.

In this course, we will not discuss the more advanced algorithms for matrix multiplication. Rather, we will give an example of standard matrix multiplication and then discuss Boolean matrix multiplication.

Example of Standard Matrix Multiplication

An example of multiplying two 3×3 matrices is shown below.

$$\begin{array}{ccccc} A & \times & B & = & C \\ \begin{bmatrix} \boxed{1} & \boxed{1} & \boxed{1} \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} & \times & \begin{bmatrix} 0 & 1 & \boxed{1} \\ 1 & 1 & \boxed{0} \\ 1 & 0 & \boxed{1} \end{bmatrix} & = & \begin{bmatrix} 2 & 2 & \boxed{2} \\ 2 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \end{array}$$

For example, to find the product term $C[1,3]$, we take row 1 of *A*, column 3 of *B*, and compute their inner product. (The terms involved in this computation are boxed.)

$$C[1,3] = A[1,1] * B[1,3] + A[1,2] * B[2,3] + A[1,3] * B[3,3] = 1 * 1 + 1 * 0 + 1 * 1 = 2$$

Boolean Matrix Multiplication

When we multiply two Boolean matrices, their product is also a Boolean matrix.

One way to do this is to first compute the product in the standard way, as was done above, and then convert the product to Boolean. That is, any product term that is non-zero is converted to 1.

$$\begin{matrix} \mathit{Standard\ Product} & & \mathit{Boolean\ Product} \\ \begin{bmatrix} 2 & 2 & 2 \\ 2 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} & \xrightarrow{yields} & \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \end{matrix}$$

Alternatively, we may compute the Boolean product directly. The formula for each product term $C[i,j]$ is obtained from the earlier formula by changing * to ∧ and + to ∨.

$$C[i,j] = \bigvee_{k=1}^{n} (A[i,k] \wedge B[k,j])$$

Below is the direct computation of the Boolean product for the above matrices. For example, the product term $C[1,3]$ is computed as follows. (The terms are boxed.)

$$C[1,3] = (A[1,1] \wedge B[1,3]) \vee (A[1,2] \wedge B[2,3]) \vee (A[1,3] \wedge B[3,3]) = 1 \wedge 1 \vee 1 \wedge 0 \vee 1 \wedge 1 = 1$$

$$\begin{matrix} A & \times & B & = & C \end{matrix}$$

$$\begin{bmatrix} \boxed{1} & \boxed{1} & \boxed{1} \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & \boxed{1} \\ 1 & 1 & \boxed{0} \\ 1 & 0 & \boxed{1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & \boxed{1} \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

A special case of matrix multiplication is when we square a matrix, $A^2 = A \times A$. In this case, the formula for computing $A^2[i,j]$ becomes:

$$A^2[i,j] = \bigvee_{k=1}^{n} (A[i,k] \wedge A[k,j])$$

Observe from the formula that the product term $A^2[i,j]$ becomes 1 if and only if there is a pair $(A[i,k], A[k,j])$ with both equal 1. Symbolically:

$$\underbrace{A^2[i,j] = 1}_{(1)} \quad \text{if and only if} \quad \underbrace{\exists k, A[i,k] = 1 \ \wedge A[k,j] = 1}_{(2)}$$

This means (1) and (2) are equivalent.

Matrix Multiplication to Test Transitivity

As we saw earlier, a relation is transitive if and only if the following is true.

$$T: \quad \forall i\, \forall j, \text{ if } \underbrace{\exists k, (A[i,k]=1 \wedge A[k,j]=1)}_{\substack{hypothesis \\ equivalent\ to \\ A^2[i,j]=1}} \quad \text{then } \underbrace{A[i,j]=1}_{conclusion}$$

We can replace the hypothesis with its equivalent, which is $(A^2[i,j]=1)$, to reach the following simple test for transitivity.

A relation *R*, with Boolean matrix *A*, is transitive if and only if the following is true.

$$T: \quad \forall i\, \forall j, \quad \text{if } \underbrace{A^2[i,j]=1}_{hypothesis} \quad \text{then } \underbrace{A[i,j]=1}_{conclusion}$$

An equivalent contrapositive form of this test is:

$$T: \quad \forall i\, \forall j, \quad \text{if } A[i,j]=0 \text{ then } A^2[i,j]=0$$

Negation of this test says the relation is NOT transitive if and only if:

$$\neg T: \quad \exists i\, \exists j, \quad (A[i,j]=0) \wedge (A^2[i,j]=1)$$

In words, the test is as follows.

- Compute the square matrix, $A^2 = A \times A$
- Compare A and A^2 term-by-term. The relation is NOT transitive if there is some i, j entry such that
$$A[i,j]=0 \text{ and } A^2[i,j]=1$$
- Otherwise, it is transitive.

Example: Consider the relation $R = \{(1,2), (3,1), (3,3)\}$, which has the following matrix.

$$A = \begin{bmatrix} 0 & \boxed{1} & 0 \\ 0 & 0 & 0 \\ \boxed{1} & \boxed{0} & 1 \end{bmatrix}$$

By direct observation of the matrix, we see the relation is not transitive, because

$$A[3,1] = 1 \wedge A[1,2] = 1, \qquad \text{but } \ A[3,2] = 0$$

(These terms are boxed.)

Now, let us use matrix multiplication to test for transitivity.

$$A^2 = A \times A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & \boxed{0} & 1 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & \boxed{1} & 1 \end{bmatrix}$$

We conclude that the relation is **not transitive** because

$$A[3,2] = 0 \ \ and \ \ A^2[3,2] = 1.$$

In words, a 0 entry in matrix *A* became a 1 in the square matrix A^2.

■

Example: Consider the relation $R = \{(1,2), (3,1), (3,2), (3,3)\}$, with the matrix

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Then,

$$A^2 = A \times A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

This relation is transitive, because for every (i, j),

$$\text{if } A[i, j] = 0 \ \text{ then } A^2[i, j] = 0.$$

Note in this example we have a 1 in the original matrix which becomes 0 in the square matrix. That is, $A[1,2] = 1 \ and \ A^2[1,2] = 0.$ But this does not negate transitivity.

■

Application: In the above two examples, we first saw a relation (matrix) that was not transitive because $A[3,2] = 0$. The second example used the same matrix, except with $A[3,2] = 1$, and that made the relation transitive. Later in this course, in the chapter on graphs, we will see how this basic process is used to compute the *transitive closure of a graph* by repeated matrix multiplications.

Relational Databases

A *database* is a large collection of records relating to a certain application, such as students registration system, and airline reservation system. A *relational database* is a model of database based on the concept of *n*-ary relation.

So far, we studied a *binary* relation, which is a set of ordered pairs, and has a table with *two* columns. A generalization is an *n*-ary *relation*, which is a set of *n*-tuples, and has a table with *n* columns. (A binary relation is a special case of *n*-ary relation, with $n = 2$.) The relation may also be considered as a set of records, where each record corresponds to each row of the table. Each column of an *n*-ary relation is called an attribute of the relation. The *key* is an attribute that uniquely identifies each *n*-tuple. A *query* is a request from a database for certain information.

In this section, we consider some specific examples of relational databases, we briefly study the basic database operations, and how queries are performed in terms of the basic operations.

Tables 1-4 are examples of *n*-ary relations. Table 1 is a relation STUDENTS with four columns (four attributes). Each row of the table is a record which relates a student's Name, to the student's ID, Major, and Advisor. The key is the ID number, which is unique for each student. Table 2 gives the students GRADES (GPA), Table 3 is the list of FACULTY, and Table 4 shows the college organization.

Table 1: STUDENTS

Name	ID	Major	Advisor
Shah	5543	CS	Yang
Malek	4822	Comp Music	Brown
Zhang	3482	Networks	Kurt
Jordan	2469	Networks	Brown
King	1999	Algorithms	Yang
Lee	7220	Materials	Lin
Carlson	9618	CS	Yang

Table 1: GRADES

SID	GPA
1999	4.0
2469	3.4
3482	3.6
4822	3.2
5543	3.7
7220	3.8
9618	4.0

Table 2: FACULTY

FName	FID	FDept
Yang	520	CS
Ming	318	IT
Kurt	255	CS
Brown	482	IS
Lin	711	ME

Table 3: COLLEGES

Dept	College
CS	Computing
IS	Computing
IT	Computing
ME	Engineering

Query

A *query* is a request for certain information. For example, a query for the relation STUDENTS may be “Find the list of all students who are advisees of Professor Yang”. We briefly discuss three basic operations in a relational database used to process queries: select, project, and join.

Select

The *select* operation selects a subset of rows (*n*-tuples), based on certain conditions. For example, for the relation STUDENTS, the following operation selects all students whose advisor is Yang.

STUDENTS [Advisor = Yang]

The result is the following relation (table).

Name	ID	Major	Advisor
Shah	5543	CS	Yang
King	1999	Algorithms	Yang
Carlson	9618	CS	Yang

Project

The *project* operation selects a subset of specified columns in a relation. The operation may result in duplicate rows, if the key is not included in the selection. In that case, duplicate rows are eliminated. For example, the following project operation on STUDENTS picks three of the columns as specified.

STUDENTS [Name, ID, Advisor]

The resulting table follows.

Name	ID	Advisor
Shah	5543	Yang
Malek	4822	Brown
Zhang	3482	Kurt
Jordan	2469	Brown
King	1999	Yang
Lee	7220	Lin
Carlson	9618	Yang

Join

The join operation combines **two** relations in a specified manner. The join operation on relations $R_1$and R_2 considers all rows in the Cartesian product $R_1 \times R_2$, which means each row of R_1 followed by each row of R_2. Each combination of the two rows that satisfies the specified condition is included as a row in the joined table.

For example, the following join operation is on relations STUDENTS and GRADES. The condition states that the ID attribute of STUDENTS must match the SID attribute of GRADES. (The attributes ID and SID result in a single attribute in the joined table.)

STUDENTS [ID = SID] GRADES

Name	**ID**	**Major**	**Advisor**	**GPA**
Shah	5543	CS	Yang	3.7
Malek	4822	Comp Music	Brown	3.2
Zhang	3482	Networks	Kurt	3.6
Jordan	2469	Networks	Brown	3.4
King	1999	Algorithms	Yang	4.0
Lee	7220	Materials	Lin	3.8
Carlson	9618	CS	Yang	4.0

Some queries may require a sequence of these basic operations to be performed.

Example: Consider the query "Find the name and ID number of all students in the College of Computing with GPA ≥ 3.5." This may be accomplished by the following sequence of operations.

1. TEMP1 := STUDENTS [ID = SID, GPA ≥ 3.5] GRADES
   (This assumes that Join and select operations may be combined in one step. Otherwise, it could be done in two steps.)

Table TEMP1

Name	**ID**	**Major**	**Advisor**	**GPA**
Shah	5543	CS	Yang	3.7
Zhang	3482	Networks	Kurt	3.6
King	1999	Algorithms	Yang	4.0
Lee	7220	Materials	Lin	3.8
Carlson	9618	CS	Yang	4.0

2. TEMP2 := FACULTY [FDept = Dept] COLLEGES

Table TEMP2

FName	FID	FDept	College
Yang	520	CS	Computing
Ming	318	IT	Computing
Kurt	255	CS	Computing
Brown	482	IS	Computing
Lin	711	ME	Engineering

3. TEMP3 := TEMP1 [Advisor = FName] TEMP2

Table TEMP3

Name	ID	Major	Advisor	GPA	FID	FDept	College
Shah	5543	CS	Yang	3.7	520	CS	Computing
Zhang	3482	Networks	Kurt	3.6	255	CS	Computing
King	1999	Algorithms	Yang	4.0	520	CS	Computing
Lee	7220	Materials	Lin	3.9	711	ME	Engineering
Carlson	9618	CS	Yang	4.0	520	CS	Computing

4. TEMP4 := TEMP3 [College = Computing]

Table TEMP4

Name	ID	Major	Advisor	GPA	FID	FDept	College
Shah	5543	CS	Yang	3.7	520	CS	Computing
Zhang	3482	Networks	Kurt	3.6	255	CS	Computing
King	1999	Algorithms	Yang	4.0	520	CS	Computing
Carlson	9618	CS	Yang	4.0	520	CS	Computing

5. TEMP5 := TEMP4 [Name, ID]

Table TEMP5

Name	ID
Shah	5543
Zhang	3482
King	1999
Carlson	9618

Finally, note that the sequence of operations for a given query may not be unique. For the query of this example, another sequence of operations is as follows.

1. TEMP1 := STUDENTS [ID = SID] GRADES
2. TEMP2 := TEMP1 [Advisor = FNAME] FACULTY
3. TEMP3 := TEMP2 [FDept = Dept] COLLEGES
4. TEMP4 := TEMP3 [GPA ≥ 3.5, College = Computing]
5. TEMP5 := TEMP4 [Name, ID]

■