# Functions and Relations

Reading 12.1, 12.2, 12.3

Recall Cartesian products and pairs: E.g. $\{1, 2, 3\} \times \{T, F\}$

$$\{(1, T), (1, F), (2, T), (2, F), (3, T), (3, F)\}$$

What is a function?

**Informal Defn.** A function is a rule that, for each member of one set (the domain), identifies a single member of another set (the range).

**Definition:** A *binary relation* $R$ consists of $3$ things

- a set $\mathrm{dom}(R)$, called its *domain*
- a set $\mathrm{rng}(R)$, called its *range*
- a set $\mathrm{graph}(R)$, called its *graph*. Such that
  - $*$ the graph is set of pairs with the first member from $\mathrm{dom}(R)$ and the second from $\mathrm{rng}(R)$. I.e.
  $$\mathrm{graph}(f) \subseteq \mathrm{dom}(R) \times \mathrm{rng}(R)$$

**Example:** $\mathrm{dom}(R) = \{1, 2, 3, 4\}$ $\mathrm{rng}(R) = \{1, 2, 3, 4\}$

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

**Example:** $\mathrm{dom}(R) = \mathrm{rng}(R) = \mathbb{R}$





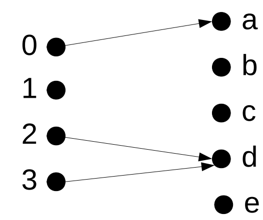
- $(x, y) \in \mathrm{graph}(R)$ iff $x^2 + y^2 = 1$.

**Notation:** We write $xRy$ to mean $(x, y) \in \mathrm{graph}(R)$. The text writes $(x, y) \in R$ to mean the same.

**Definition:** A *partial function* $f$ is a relation such that each member of the domain appears at most once as the first member of a pair in the graph:

$(x, y_0) \in \mathrm{graph}(f) \wedge (x, y_1) \in \mathrm{graph}(f) \longrightarrow y_0 = y_1$, for all $x, y_0, y_1$

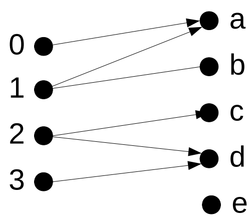
**Example:** A relation that is a partial function:




**Definition:** A *total relation* $f$ is a relation such that each member of the domain appears at least once as the first member of a pair in the graph:

$$\forall x \in \mathrm{dom}(f), \exists y \in \mathrm{rng}(f), (x, y) \in \mathrm{graph}(f)$$

**Example:** A relation that is a total relation:




**Definition:** A *function* $f$ is a relation such that each member of the domain appears at exactly once as the first member of a pair in the graph.

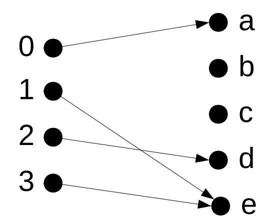
This last requirement can be formalized into two parts

- $f$ is a partial function
- $f$ is a total relation.

**Example:** A relation that is both a partial function and a total relation:








**Note:** Every function is a partial function and every partial function is a relation.

**Notation:**

- We write $f : D \to R$ to mean $f$ is a function with $\mathrm{dom}(f) = D$ and $\mathrm{rng}(f) = R$.
- We write $f : D \rightsquigarrow R$ to mean $f$ is a partial function with $\mathrm{dom}(f) = D$ and $\mathrm{rng}(f) = R$.
- And if $f$ is a partial function or a function, we write $f(x) = y$ to mean $(x, y) \in \mathrm{graph}(f)$

**Note:** The text does not mention the $\mathrm{graph}$ and simply writes $(x, y) \in R$ where I'm writing $(x, y) \in \mathrm{graph}(R)$.

**Example:** function

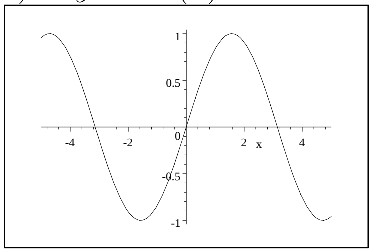
- $f1 : \{0, 1, 2, 3\} \to \{T, F\}$
- $\mathrm{graph}(f1) = \{(0, T), (1, F), (2, T), (3, F)\}$

**Example:** function

- $f2 : \{0, 1, 2, 3\} \to \{0, 1, ..., 6\}$
- $\mathrm{graph}(f2) = \{(0, 0), (1, 2), (2, 4), (3, 6)\}$

**Example:** function

- $\sin : \mathbb{R} \to \mathbb{R}$
- $(x, y) \in \mathrm{graph}(\sin)$ iff $y = \sin(x)$




**Example:** relation

- $\mathrm{dom}(f3) = \{0, 1, 2, 3\}$, $\mathrm{rng}(f3) = \{T, F\}$
- $\mathrm{graph}(f3) = \{(0, T), (1, F), (2, T), (3, F), (0, F)\}$

**Example:** partial function

- $f4 : \{0, 1, 2, 3\} \rightsquigarrow \{0, 1, ..., 6\}$
- $\mathrm{graph}(f4) = \{(0, 0), (1, 2), (2, 4)\}$



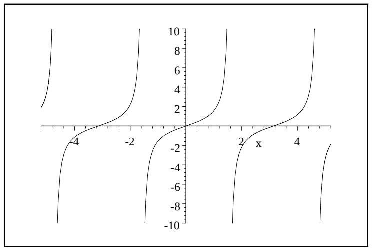


Example: function

- $f5 : (-\pi/2, \pi/2) \to \mathbb{R}$
- $f5(x) = \tan(x)$

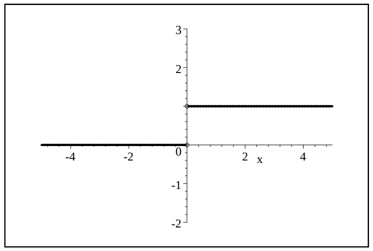
**Example:** partial function

- $\tan : \mathbb{R} \rightsquigarrow \mathbb{R}$




**Example:** partial function. The step function.

- $f6 : \mathbb{R} \rightsquigarrow \mathbb{R}$
- $\mathrm{graph}(f6) = \{(x, 0) \mid x \in \mathbb{R} \wedge x < 0\} \cup \{(x, 1) \mid x \in \mathbb{R} \wedge x > 0\}$




## Inversion, one-one, and onto

**Definition:** The *inverse* of a relation $R$ is a relation $R^{-1}$ such that

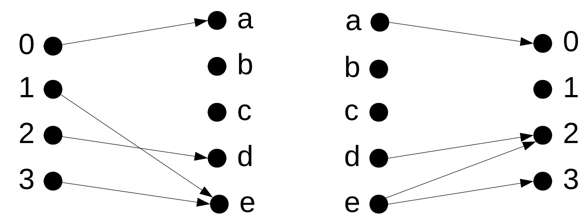
- $\mathrm{dom}(R^{-1}) = \mathrm{rng}(R)$
- $\mathrm{rng}(R^{-1}) = \mathrm{dom}(R)$
- $\mathrm{graph}(R^{-1}) = \{(y, x) \mid (x, y) \in \mathrm{graph}(R)\}$

Note that $\left(R^{-1}\right)^{-1} = R$, for all relations $R$.





**Example:**




**Example:** Consider the relation $P$ for parent. $xPy$ if $x$ is $y$'s parent

- Consider $C = P^{-1}$
- Then $yCx$ is true only if $x$ is $y$'s parent
- What is $C$ in English?

Note that the inverse of a function may or may not be a function.

**Example:** Consider

- $f1 : \{0, 1, 2, 3\} \rightarrow \{T, F\}$
- $\mathrm{graph}(f1) = \{(0, T), (1, F), (2, T), (3, F)\}$
- Then $\mathrm{graph}(f1^{-1}) = \{(T, 0), (T, 2), (F, 1), (F, 3)\}$
- This can not be the graph of a function, since $T$ (for example) occurs twice as a the first item of a pair.

**Example:** Consider

- $f2 : \{0, 1, 2, 3\} \to \{0, 1, ..., 6\}$
- $\text{graph}(f2) = \{(0, 0), (1, 2), (2, 4), (3, 6)\}$
- Then $\text{graph}(f2^{-1})$ is $\{(0, 0), (2, 1), (4, 2), (6, 3)\}$. But the domain of $f2^{-1}$ is $\{0, 1, ..., 6\}$ so the $1$ (for example) does not occur as the first member of a pair.
- $f2^{-1}$ is a partial function.

Which relations have inverses that are functions?

**Definition:** A relation is *one-one* if every member of the range appears at most once as the second member of some pair in the graph.

**Theorem:**

- The inverse of a one-one relation is a partial function.
- The inverse of a partial function is a one-one relation.

**Definition:** A relation is *onto* if every member of the range appears at least once as the second member of some pair in the graph.

**Theorem:**

- The inverse of an onto relation is a total relation.



Discrete Math. for Engineering, 2005. Notes 8. Functions and Relations © Theodore Norvell, Memorial University

- The inverse of a total relation is an onto relation.

**Theorem:**

- The inverse of a one-one and onto relation is a function.
- And the inverse of a function is a one-one and onto relation.

**Corollary:** The inverse of a one-one and onto function is a one-one and onto function.

**Example:**

- $f7 : \mathbb{Z} \to \mathbb{Z}$, $\text{graph}(f7) = \{(n, n + 10) \mid n \in \mathbb{Z}\}$
- This function is one-one and onto.
- Its inverse is a function $f7^{-1} : \mathbb{Z} \to \mathbb{Z}$, $\text{graph}(f7^{-1}) = \{(n, n - 10) \mid n \in \mathbb{Z}\}$

**Example:** Consider a function from $16$ bit strings to $16$ bit strings which swaps the first and second byte of the string

- $swap : \{F, T\}^{16} \to \{F, T\}^{16}$

- $$swap(\langle b_{15}, b_{14}, b_{13}, b_{12}, b_{11}, b_{10}, b_9, b_8, b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\rangle)$$
$$= \langle b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0, b_{15}, b_{14}, b_{13}, b_{12}, b_{11}, b_{10}, b_9, b_8\rangle$$
- This one-one onto function is its own inverse. $swap^{-1} = swap$.

## Identity and composition

*Identity function.* For each set $A$, the function $id_A : A \to A$ maps each element or $A$ to itself.

$$id_A(x) = x, \text{ for all } x \in A$$

*Composition.*

Consider the relation $P$ for parent. $xPy$ iff $x$ is $y$'s parent

- Define a relation $Q$ so that $xQy$ iff there is a $z$ such that $zPx$ and $zPy$.
- What is $Q$ in English?

Consider the relation $xQy$ meaning $x$ is $y$'s sibling

- Define relation $K$ so that $xKy$ iff there is are $w$ and $z$ such that $wPy$ and $wQz$ and $zPx$.
- What is $K$ in English?





**Defn:** Suppose $\mathrm{rng}(R) = \mathrm{dom}(S)$. The *composition of $S$ following $R$*, written $S \circ R$ is a relation such that

- $\mathrm{dom}(S \circ R) = \mathrm{dom}(R)$
- $\mathrm{rng}(S \circ R) = \mathrm{rng}(S)$
- $\mathrm{graph}(S \circ R)$ is such that

$(x\,(S \circ R)\,y$ iff $\exists z, xRz \wedge zSy)$ , for all $x \in \mathrm{dom}(R), y \in \mathrm{rng}(S)$

**Example:** $Q = P \circ P^{-1}$

**Example:** $K = P \circ Q \circ P^{-1}$

**Example:** Suppose that $f$ and $g$ are functions, then

$$(f \circ g)(x) = f(g(x)), \text{ for all } x \in \mathrm{dom}(g)$$

Note that $\circ$ is associative and has identity $id$ and the empty relation is a dominator.

$$\begin{aligned} T \circ (S \circ R) &= (T \circ S) \circ R \\ R \circ id &= R = id \circ R \\ R \circ \emptyset &= \emptyset = \emptyset \circ R \end{aligned}$$

In general $\circ$ is not commutative, nor is it idempotent.

$S \circ R$ may not equal $R \circ S$

$R \circ R$ may not equal $R$

Suppose that a relation $R$ has $\mathrm{dom}(R) = \mathrm{rng}(R) = A$.

- Then $R^0$ is $id_A$
- $R^1 = R$
- $R^2 = R \circ R$
- $R^3 = R \circ R \circ R$
- Etc.

**Example:** Suppose that $xRy$ means that two nodes in a network are directly connected (1 hop)

- Then $x(R \circ R)y$ means that $x$ and $y$ are connected by 2 hops.
- and $id \cup R \cup (R \circ R)$ means[1] that 2 nodes are connected by $0$, $1$, or $2$ hops.
- Define $R^0 = id$, $R^1 = R$, $R^n = (R \circ R^{n-1})$ for $n \geq 1$
- Then $R^0 \cup R^1 \cup R^2 \cup \cdots$ is a relation that indicates whether two computers are connected by any number of hops.
- This is called the reflexive and transitive closure of $R$.
- The notation is $R^*$

[1] The union of relations is the relation formed by unioning the domains, ranges, and graphs.





We can compute the reflexive and transitive closure of $R$ as follows

---

$T := id_A$ ; // Where $\mathrm{dom}(R) = \mathrm{rng}(R) = A$
$U := id_A$
$i := 0$ ;
// Invariant: $T = \bigcup_{j \in \{0,1,\ldots,i\}} R^j$ and $U = R^i$
while( true ) {
  $U := U \circ R$ ;
  if( $U \subseteq T$ ) break ;
  $T := U \cup T$ ;
  $i := i + 1$ }

---

This is very useful, for example, to determine if a network is fully connected.

# Relational Databases

Currently most database management systems are based on the "relational model".

Examples include, Access, Oracle, and MySQL.

## Tables and Databases

A *table* (or $n$-ary relation) $R$ has

- A tuple of $n$ distinct attribute names $\text{attr}(R) = (c_0, c_1, ...c_{n-1})$
- $n$ domain sets $\text{dom}(R) = (D_0, D_1, \cdots, D_{n-1})$
- $\text{graph}(R) \subseteq D_0 \times D_1 \times \cdots \times D_{n-1}$

We can visualize a table as a matrix in which

- each column has a name and is associated with a set of potential values
- no row is repeated
- the order of the rows does not matter





**Examples:**

Personnel

| personnel-num | name | salary | boss |
|---|---|---|---|
| 001 | Sue King | 100000 | 001 |
| 002 | Fong Ping | 40000 | 001 |
| 999 | Bob Will | 20000 | 001 |

Projects:

| Name | Assigned | Completion-date |
|---|---|---|
| Snipe | 001 | 2003-12-31 |
| Snipe | 999 | 2003-12-31 |
| Snark | 999 | 2004-01-31 |

A *relational database* is

- a set of $m$ table names $\{t_0, t_1, ..., t_{m-1}\}$
- $m$ tables indexed by name $T_{t_0}, T_{t_1}, ..., T_{t_{m-1}}$

Example: The set of table names is $\{personnel, projects\}$ and the tables $T_{personnel}$ and $T_{projects}$ are the tables above.

## Query operations on data bases

Query operations: projection, attribute renaming, selection, join.

**Projection:**

- Given a tuple $p = (v_0, v_1, \cdots v_{n-1})$ from a table $T$ with attributes $(c_0, c_1, ...c_{n-1})$. Consider a sequence of distinct attributes $a' = (c_{i_0}, c_{i_1}, ..., c_{i_{k-1}})$
  - ∗ define the *projection* of $p$ onto $a'$ (written $p[(c_{i_0}, c_{i_1}, ..., c_{i_{k-1}})]$) to be the tuple $(v_{i_0}, v_{i_1}, ..., v_{i_{k-1}})$
- For a table $T$ define the *projection* of $T$ onto $a'$ as a table $T'$ with
  - ∗ attributes $a'$
  - ∗ domains $(D_{i_0}, D_{i_1}, \cdots, D_{i_{k-1}})$
  - ∗ graph
$$\{p[(c_{i_0}, c_{i_1}, ..., c_{i_{k-1}})] \mid p \in \mathrm{graph}(R)\}$$

**Example:** If we want to know who works for whom, but hide salary information, we can project out the salary:

- Personnel[personnel-num, name, boss]

Suppose we want to know who has a management position:

- Personnel[boss] gives

| **boss** |
|---|
| 001 |





**Attribute Renaming.**

- Sometimes we need to rename the attributes. We can combine this with projection. E.g.
- Projects[name ⇝ project-name, assigned ⇝ personnel-num]
- This is the same table as Projects[name, assigned], except with different attribute names.

**Selection:**

- Suppose $T$ is a table with attributes $(c_0, c_1, ...c_{n-1})$ and $E$ is a boolean expression with variable names drawn from $\{c_0, c_1, ...c_{n-1}\}$. Then
$$T \mid E$$
is a table with attributes and domains the same as $T$ and graph
$$\{(c_0, c_1, ...c_{n-1}) \in \mathrm{graph}(T) \mid E\}$$
- Example: suppose we want to know all the personnel making more than 50000
$$\text{Personnel} \mid \text{salary} > 50000$$
- Example: Bob wants to know the names of all his

projects due this year

(projects | assigned=999 $\wedge$ completion-date $<$ 2004-01-01)
[name]

**Join:**

- Join combines two tables.
- Consider tables
  - Names

| **student-num** | **name** |
|---|---|
| 12345 | Smith |
| 23456 | Jones |
| 11235 | Seth |
| 31415 | Lee |

and

  - Marks

| **student-num** | **mark** |
|---|---|
| 12345 | A+ |
| 23456 | B |
| 11235 | B+ |
| 31415 | F |

- Then the join Names*Marks is





- Suppose $A$ and $B$ are tables with attribute names
$$\mathrm{attr}(A) = (a_0, a_1, ...a_{m-1})$$
$$\mathrm{attr}(B) = (b_0, b_1, ..., b_{n-1})$$
and domains
$$\mathrm{dom}(A) = (A_0, A_1, ..., A_{m-1})$$
$$\mathrm{dom}(B) = (B_0, B_1, ..., B_{n-1})$$
- We say $A$ and $B$ are *join-compatible* iff equally named attributes correspond to equal domains. I.e. iff $a_j = b_k$ implies $A_j = B_k$ (for all $j$, $k$)
- The *join* of join-compatible tables $A$ and $B$, $A * B$, is a table $C$ such that
  - the set of attributes is the union of the sets of attributes of $A$ and $B$
    i.e. if
$$\mathrm{attr}(C) = (c_0, c_1, ...c_{p-1})$$

then

$\{c_0, c_1, ..., c_{p-1}\} = \{a_0, ...a_{n-1}\} \cup \{b_0, ..., b_{m-1}\}$

- $*$ the domains correspond to the domains in $A$ and $B$. I.e. if

$$\mathrm{dom}(C) = (C_0, ...C_{p-1})$$

then (for all $i, j, k$) if $c_i = a_j$ then $C_i = A_j$ and if $c_i = b_k$ then $C_i = B_k$.

- $*$ The graph consists of tuples that combine the values from tuples in $A$ and $B$.
- $*$ I.e. $x$ is a tuple of $C$ iff there exist tuples $y$ from $A$ and $z$ from $B$ such that

$$x[\mathrm{attr}(A)] = y$$

and

$$x[\mathrm{attr}(B)] = z$$

- $*$ Note that $y$ and $z$ must agree on the values of any common attributes.

- Example: I want to know the names of people assigned to various projects

Projects[name $\rightsquigarrow$ project-name,assigned $\rightsquigarrow$ personnel-num]
* Personnel[personnel-num,name]

- Gives





- How do we make this table?

| **personnel-num** | **name** | **boss** | **boss-name** |
|---|---|---|---|
| 001 | Sue King | 001 | Sue King |
| 002 | Fong Ping | 001 | Sue King |
| 999 | Bob Willing | 001 | Sue King |

Note that if we have binary relations then composition is essentially a join followed by a projection. I.e. if we regard a binary relation as a table having attributes *left* and *right*.

$S \circ R$ is $(S[$*left* $\rightsquigarrow$ *middle,right*$] * R[$*left*$,$ *right* $\rightsquigarrow$ *middle*$])[$*left*$,$ *right*$]$

SQL

- SQL is the standard (and most popular) data-base query language. It is based (loosely) on the query operations presented above.