# Q LANGUAGE - DICTIONARIES

 Copyright © tutorialspoint.com

Dictionaries are an extension of lists which provide the foundation for creating tables. In mathematical terms, dictionary creates the

“domain → Range”

or in general *short* creates

“key → value”

relationship between elements.

A dictionary is an ordered collection of key-value pairs that is roughly equivalent to a hash table. A dictionary is a mapping defined by an explicit I/O association between a domain list and a range list via positional correspondence. The creation of a dictionary uses the "xkey" primitive !

```
                    ListOfDomain ! ListOfRange
```

The most basic dictionary maps a simple list to a simple list.

| **Input *I*** | **Output *O*** |
|---|---|
| `Name | `John |
| `Age | 36 |
| `Sex | “M” |
| Weight | 60.3 |

```
q)d:`Name`Age`Sex`Weight!(`John;36;"M";60.3)   / Create a dictionary d

q)d

Name   | `John
Age    | 36
Sex    | "M"
Weight | 60.3

q)count d              / To get the number of rows in a dictionary.
4

q)key d                / The function key returns the domain
`Name`Age`Sex`Weight

q)value d              / The function value returns the range.

`John
36

"M"
60.3

q)cols d             / The function cols also returns the domain.
`Name`Age`Sex`Weight
```

## Lookup

Finding the dictionary output value corresponding to an input value is called **looking up** the input.

```
q)d[`Name]       / Accessing the value of domain `Name
`John

q)d[`Name`Sex]   / extended item-wise to a simple list of keys
`John
"M"
```

## Lookup with Verb @

```
q)d1:`one`two`three!9 18 27

q)d1[`two]
18

q)d1@`two
18
```

## Operations on Dictionaries

## Amend and Upsert

As with lists, the items of a dictionary can be modified via indexed assignment.

```
d:`Name`Age`Sex`Weight! (`John;36;"M";60.3)
                                  / A dictionary d

q)d[`Age]:35                      / Assigning new value to key Age

q)d
                                / New value assigned to key Age in d
Name   | `John
Age    | 35
Sex    | "M"
Weight | 60.3
```

Dictionaries can be extended via index assignment.

```
q)d[`Height]:"182 Ft"

q)d

Name   | `John
Age    | 35
Sex    | "M"
Weight | 60.3
Height | "182 Ft"
```

## Reverse Lookup with Find ?

The find ? operator is used to perform reverse lookup by mapping a range of elements to its domain element.

```
q)d2:`x`y`z!99 88 77

q)d2?77
`z
```

In case the elements of a list is not unique, the **find** returns the first item mapping to it from the domain list.

## Removing Entries

To remove an entry from a dictionary, the **delete** _ **function** is used. The left operand of _ is the

dictionary and the right operand is a key value.

```
q)d2:`x`y`z!99 88 77

q)d2 _`z

x| 99
y| 88
```

Whitespace is required to the left of _ if the first operand is a variable.

```
q)`x`y _ d2           / Deleting multiple entries

z| 77
```

## Column Dictionaries

Column dictionaries are the basics for creation of tables. Consider the following example −

```
q)scores: `name`id!(`John`Jenny`Jonathan;9 18 27)
                                / Dictionary scores

q)scores[`name]                / The values for the name column are
`John`Jenny`Jonathan

q)scores.name                  / Retrieving the values for a column in a
                                / column dictionary using dot notation.
`John`Jenny`Jonathan

q)scores[`name][1]             / Values in row 1 of the name column
`Jenny

q)scores[`id][2]               / Values in row 2 of the id column is
27
```

## Flipping a Dictionary

The net effect of flipping a column dictionary is simply reversing the order of the indices. This is logically equivalent to transposing the rows and columns.

## Flip on a Column Dictionary

The transpose of a dictionary is obtained by applying the unary flip operator. Take a look at the following example −

```
q)scores

name  | John Jenny Jonathan
id    | 9    18    27

q)flip scores

  name     id
---------------
  John     9
  Jenny    18
 Jonathan  27
```

## Flip of a Flipped Column Dictionary

If you transpose a dictionary twice, you obtain the original dictionary,

```
q)scores ~ flip flip scores
1b
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js