

# The Data Frame I/O Reference Manual

---

Common Lisp library for reading and writing data-frames, version 2.0

Steve Nunez <steve@symbolics.tech>  
Tamas Papp <tkpapp@gmail.com>

---

# Table of Contents

<b>1</b>	<b>Systems .....</b>	<b>1</b>
1.1	dfio.....	1
<b>2</b>	<b>Files .....</b>	<b>3</b>
2.1	Lisp.....	3
2.1.1	dfio.asd.....	3
2.1.2	dfio/pkgdcl.lisp.....	3
2.1.3	dfio/decimal.lisp.....	3
2.1.4	dfio/string-table.lisp.....	3
2.1.5	dfio/data-column.lisp.....	4
2.1.6	dfio/utils.lisp.....	4
2.1.7	dfio/write.lisp.....	4
2.1.8	dfio/delimited-text.lisp.....	5
<b>3</b>	<b>Packages .....</b>	<b>7</b>
3.1	dfio.data-column.....	7
3.2	dfio.....	7
3.3	dfio.string-table.....	8
3.4	dfio.decimal.....	8
<b>4</b>	<b>Definitions .....</b>	<b>9</b>
4.1	Exported definitions.....	9
4.1.1	Macros.....	9
4.1.2	Functions.....	9
4.1.3	Conditions.....	12
4.1.4	Structures.....	13
4.1.5	Classes.....	13
4.2	Internal definitions.....	14
4.2.1	Special variables.....	14
4.2.2	Functions.....	14
4.2.3	Types.....	15
<b>Appendix A</b>	<b>Indexes .....</b>	<b>17</b>
A.1	Concepts.....	17
A.2	Functions.....	18
A.3	Variables.....	19
A.4	Data types.....	20



# 1 Systems

The main system appears first, followed by any subsystem dependency.

## 1.1 dfio

### Maintainer

Steve Nunez <steve@symbolics.tech>

### Author

Tamas Papp <tkpapp@gmail.com>

### License

MS-PL

### Description

Common Lisp library for reading and writing data-frames

### Version

2.0

### Dependencies

- alexandria
- anaphora
- cl-csv
- data-frame
- let-plus

### Source

[dfio.asd], page 3, (file)

### Directory

s:/src/dfio/

### Components

- [pkgdcl.lisp], page 3, (file)
- [decimal.lisp], page 3, (file)
- [string-table.lisp], page 3, (file)
- [data-column.lisp], page 4, (file)
- [utils.lisp], page 4, (file)
- [write.lisp], page 4, (file)
- [delimited-text.lisp], page 5, (file)



## 2 Files

Files are sorted by type and then listed depth-first from the systems components trees.

### 2.1 Lisp

#### 2.1.1 dfio.asd

**Location** dfio.asd

**Systems** [dfio], page 1, (system)

#### 2.1.2 dfio/pkgdcl.lisp

**Parent** [dfio], page 1, (system)

**Location** pkgdcl.lisp

**Packages**

- [dfio.data-column], page 7,
- [dfio], page 7,
- [dfio.string-table], page 8,
- [dfio.decimal], page 8,

#### 2.1.3 dfio/decimal.lisp

**Dependency**

[pkgdcl.lisp], page 3, (file)

**Parent** [dfio], page 1, (system)

**Location** decimal.lisp

**Exported Definitions**

- [parse-rational], page 9, (function)
- [parse-rational-error], page 12, (condition)
- [parse-real], page 10, (function)

**Internal Definitions**

- [+exponent-chars+], page 14, (special variable)
- [gobble-positive-integer], page 15, (function)
- [gobble-sign], page 15, (function)

#### 2.1.4 dfio/string-table.lisp

**Dependency**

[decimal.lisp], page 3, (file)

**Parent** [dfio], page 1, (system)

**Location** string-table.lisp

**Exported Definitions**

- [string-table], page 10, (function)
- [string-table], page 13, (structure)
- [string-table-add], page 11, (function)
- [string-table-count], page 11, (function)

- [string-table-duplicate], page 12, (condition)
- [string-table-intern], page 11, (function)
- [string-table-lookup], page 11, (function)
- [string-table-not-found], page 13, (condition)
- [string-table-strings], page 11, (function)

#### Internal Definitions

- [copy-string-table], page 14, (function)
- [string-table-get], page 15, (function)
- [(setf string-table-get)], page 15, (function)
- [string-table-p], page 15, (function)
- [string-table-table], page 15, (function)
- [(setf string-table-table)], page 15, (function)

### 2.1.5 dfio/data-column.lisp

#### Dependency

[string-table.lisp], page 3, (file)

**Parent** [dfio], page 1, (system)

**Location** data-column.lisp

#### Exported Definitions

- [data-column], page 9, (function)
- [data-column], page 13, (class)
- [data-column-add], page 9, (function)
- [data-column-counts], page 9, (function)
- [data-column-vector], page 9, (function)

#### Internal Definitions

[non-negative-integer], page 15, (type)

### 2.1.6 dfio/utils.lisp

#### Dependency

[data-column.lisp], page 4, (file)

**Parent** [dfio], page 1, (system)

**Location** utils.lisp

#### Exported Definitions

- [string-to-keyword], page 11, (function)
- [string-to-symbol], page 11, (function)

### 2.1.7 dfio/write.lisp

#### Dependency

[utils.lisp], page 4, (file)

**Parent** [dfio], page 1, (system)

**Location** write.lisp

#### Exported Definitions

- [save], page 9, (macro)
- [write-df], page 9, (macro)

### 2.1.8 dfio/delimited-text.lisp

**Dependency**

[write.lisp], page 4, (file)

**Parent**

[dfio], page 1, (system)

**Location**

delimited-text.lisp

**Exported Definitions**

- [read-csv], page 10, (function)
- [write-csv], page 12, (function)

**Internal Definitions**

[csv-to-data-columns], page 14, (function)





## 3 Packages

Packages are listed by definition order.

### 3.1 dfio.data-column

**Source** [pkgdcl.lisp], page 3, (file)

**Use List**

- let-plus
- [dfio.string-table], page 8,
- [dfio.decimal], page 8,
- anaphora
- common-lisp

**Used By List**

[dfio], page 7,

**Exported Definitions**

- [data-column], page 9, (function)
- [data-column], page 13, (class)
- [data-column-add], page 9, (function)
- [data-column-counts], page 9, (function)
- [data-column-vector], page 9, (function)

**Internal Definitions**

[non-negative-integer], page 15, (type)

### 3.2 dfio

**Source** [pkgdcl.lisp], page 3, (file)

**Use List**

- [dfio.data-column], page 7,
- let-plus
- anaphora
- alexandria
- common-lisp

**Exported Definitions**

- [read-csv], page 10, (function)
- [save], page 9, (macro)
- [string-to-keyword], page 11, (function)
- [string-to-symbol], page 11, (function)
- [write-csv], page 12, (function)
- [write-df], page 9, (macro)

**Internal Definitions**

[csv-to-data-columns], page 14, (function)

### 3.3 dfio.string-table

**Source** [pkgdcl.lisp], page 3, (file)

**Use List**

- let-plus
- anaphora
- alexandria
- common-lisp

**Used By List**

[dfio.data-column], page 7,

**Exported Definitions**

- [string-table], page 10, (function)
- [string-table], page 13, (structure)
- [string-table-add], page 11, (function)
- [string-table-count], page 11, (function)
- [string-table-duplicate], page 12, (condition)
- [string-table-intern], page 11, (function)
- [string-table-lookup], page 11, (function)
- [string-table-not-found], page 13, (condition)
- [string-table-strings], page 11, (function)

**Internal Definitions**

- [copy-string-table], page 14, (function)
- [string-table-get], page 15, (function)
- [(setf string-table-get)], page 15, (function)
- [string-table-p], page 15, (function)
- [string-table-table], page 15, (function)
- [(setf string-table-table)], page 15, (function)

### 3.4 dfio.decimal

**Source** [pkgdcl.lisp], page 3, (file)

**Use List**

- let-plus
- anaphora
- common-lisp

**Used By List**

[dfio.data-column], page 7,

**Exported Definitions**

- [parse-rational], page 9, (function)
- [parse-rational-error], page 12, (condition)
- [parse-real], page 10, (function)

**Internal Definitions**

- [+exponent-chars+], page 14, (special variable)
- [gobble-positive-integer], page 15, (function)
- [gobble-sign], page 15, (function)

## 4 Definitions

Definitions are sorted by export status, category, package, and then by lexicographic order.

### 4.1 Exported definitions

#### 4.1.1 Macros

**save** *DF PATHSPEC* [Macro]

Save DF in the file named by PATHSPEC

**Package** [dfio], page 7,

**Source** [write.lisp], page 4, (file)

**write-df** *DF STREAM* [Macro]

Write DF to STREAM in a format suitable for reading back in with the Lisp reader

**Package** [dfio], page 7,

**Source** [write.lisp], page 4, (file)

#### 4.1.2 Functions

**data-column** *&key MAP-ALIST DEFAULT-FLOAT-FORMAT* [Function]

**Package** [dfio.data-column], page 7,

**Source** [data-column.lisp], page 4, (file)

**data-column-add** *DATA-COLUMN STRING* [Function]

**Package** [dfio.data-column], page 7,

**Source** [data-column.lisp], page 4, (file)

**data-column-counts** *DATA-COLUMN* [Function]

Return the counts.

**Package** [dfio.data-column], page 7,

**Source** [data-column.lisp], page 4, (file)

**data-column-vector** *DATA-COLUMN* [Function]

Return the collected elements as a vector.

**Package** [dfio.data-column], page 7,

**Source** [data-column.lisp], page 4, (file)

**parse-rational** *STRING &key START END EXPONENT-CHARS* [Function]

Parse a decimal rational in (subseq string start end) of the form  
[sign][whole].[fraction]][exponent] where

sign ::= + | - | empty

whole ::= digit\*

fraction ::= digit\*

exponent ::= exponent-char[sign]digit+

with the restriction that WHOLE and FRACTION cannot be empty at the same time. EXPONENT-CHAR is a string and contains the valid exponent chars.

Whitespace is NOT trimmed, and leads to an error. In case of a parsing failure, PARSE-RATIONAL-ERROR is used.

Return (values NUMBER DECIMAL-DOT? EXPONENT-CHAR). NUMBER is a RATIONAL, DECIMAL-DOT? is T when a decimal dot is present, otherwise NIL, EXPONENT-CHAR contains the exponent character, NIL if not present.

Numbers of the form .112 and 112. are valid syntax, representing 0.112 and 112.0, respectively.

Examples:

```
(parse-rational "7") => (values 7 NIL NIL)
(parse-rational "7.") => (values 7 T NIL)
(parse-rational "0.7") => (values 7/10 T NIL)
(parse-rational ".7") => (values 7/10 T NIL)
(parse-rational "7.e2") => (values 700 T #e)
(parse-rational ".7d1") => (values 7 T #d)
```

**Package** [dfio.decimal], page 8,

**Source** [decimal.lisp], page 3, (file)

**parse-real** *STRING &key START END S-FLOAT F-FLOAT* [Function]  
*D-FLOAT L-FLOAT E-FLOAT*

Wrapper for PARSE-RATIONAL, converting non-integers to floats. The float type is determined by the -float arguments for each exponent character. Integers are not converted to floats. Return a single value, type of (or integer float).

See PARSE-RATIONAL for accepted formats, errors, etc.

**Package** [dfio.decimal], page 8,

**Source** [decimal.lisp], page 3, (file)

**read-csv** *STREAM-OR-STRING &key SKIP-FIRST-ROW?* [Function]  
*COLUMN-KEYS-OR-FUNCTION PACKAGE MAP-ALIST*

Read a CSV file, stream, or string into a DATA-FRAME, which is returned.

When SKIP-FIRST-ROW?, the first row is read separately and COLUMN-KEYS-OR-FUNCTION is used to form column keys.

When COLUMN-KEYS-OR-FUNCTION is a sequence, it is used for column keys, regardless of the value of SKIP-FIRST-ROW?.

PACKAGE indicates the package to intern column names into.

MAP-ALIST maps values during the import. This is useful if you want special mappings for missing, though the mechanism is general.

**Package** [dfio], page 7,

**Source** [delimited-text.lisp], page 5, (file)

**string-table** *&key (TABLE TABLE)* [Function]

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**string-table-add** *STRING-TABLE STRING &optional VALUE* [Function]  
Add *STRING* mapped to *VALUE* to *STRING-TABLE*, raising *STRING-TABLE-DUPLICATE* if *STRING* is already in the table. Return *VALUE*.

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**string-table-count** *STRING-TABLE* [Function]  
Number of distinct strings in the table.

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**string-table-intern** *STRING-TABLE STRING &optional NEW-VALUE* [Function]  
If *STRING* is already in *STRING-TABLE*, return its value, otherwise add it and return *NEW-VALUE*. When used with the default argument for *NEW-VALUE*, *EQUAL* strings are always mapped to values that are *EQ*.

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**string-table-lookup** *STRING-TABLE STRING* [Function]  
Return the value corresponding to *STRING* in *STRING-TABLE*, or raise the *STRING-TABLE-NOT-FOUND* error.

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**string-table-strings** *STRING-TABLE* [Function]  
List of strings in *STRING-TABLE*.

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**string-to-keyword** *STRING* [Function]  
Map string to a keyword.

The current implementation replaces *#.* and *#space* with a *#-*, and upcases all other characters.

**Package** [dfio], page 7,

**Source** [utils.lisp], page 4, (file)

**string-to-symbol** *STRING* [Function]  
Map *STRING* to a symbol in *PACKAGE*, replacing *#.*, *#-* and *#space* with a *#-*, and upcasing all other characters. Exports *symbol*.

**Package** [dfio], page 7,

**Source** [utils.lisp], page 4, (file)

**write-csv** *DF &key STREAM ADD-FIRST-ROW (SEPARATOR SEPARATOR) (QUOTE QUOTE) (ESCAPE QUOTE-ESCAPE) (NEWLINE WRITE-NEWLINE) (ALWAYS-QUOTE ALWAYS-QUOTE)* [Function]

Write a data-frame to a stream.

Keywords:

stream: stream to write to. Default: nil.

nil - writes the rows to a string and returns it  
an open stream

a pathname (overwrites if the file exists)

quote: quoting character. Defaults to \*quote\*

escape: escaping character. Defaults to \*quote-escape\*

newline: newline character. Defaults to \*write-newline\*

always-quote: Defaults to \*always-quote\*

add-first-row: Add column names as the first

Notes:

The :newline keyword requires a sequence, so use :newline '(#newline) or use cl-interpol

**Package** [dfio], page 7,

**Source** [delimited-text.lisp], page 5, (file)

### 4.1.3 Conditions

**parse-rational-error** () [Condition]  
Error used by parse-rational and parse-real.

**Package** [dfio.decimal], page 8,

**Source** [decimal.lisp], page 3, (file)

**Direct superclasses**  
error (condition)

**Direct slots**

string [Slot]

**Initargs** :string

**Initform** (quote nil)

message [Slot]

**Initargs** :message

**Initform** (quote nil)

**string-table-duplicate** () [Condition]  
String is already in the table.

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**Direct superclasses**  
error (condition)

**string-table-not-found ()** [Condition]  
String not found in table.

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**Direct superclasses**  
error (condition)

#### 4.1.4 Structures

**string-table ()** [Structure]  
A table of distinct strings, optionally mapping each one to a value.

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**Direct superclasses**  
structure-object (structure)

**Direct methods**  
print-object (method)

**Direct slots**

<b>table</b>	[Slot]
<b>Type</b>	hash-table
<b>Initform</b>	(make-hash-table :test (function equalp))
<b>Readers</b>	[string-table-table], page 15, (function)
<b>Writers</b>	[(setf string-table-table)], page 15, (function)

#### 4.1.5 Classes

**data-column ()** [Class]

**Package** [dfio.data-column], page 7,

**Source** [data-column.lisp], page 4, (file)

**Direct superclasses**  
standard-object (class)

**Direct slots**

<b>reverse-elements</b>	[Slot]
<b>Type</b>	list
<b>default-float-format</b>	[Slot]
<b>Type</b>	symbol
<b>Initargs</b>	:default-float-format
<b>float-count</b>	[Slot]
<b>Type</b>	dfio.data-column::non-negative-integer
<b>Initform</b>	0



<code>integer-count</code>	[Slot]
<b>Type</b> <code>dfio.data-column::non-negative-integer</code>	
<b>Initform</b> <code>0</code>	
<code>integer-min</code>	[Slot]
<b>Type</b> <code>integer</code>	
<b>Initform</b> <code>0</code>	
<code>integer-max</code>	[Slot]
<b>Type</b> <code>integer</code>	
<b>Initform</b> <code>0</code>	
<code>map-count</code>	[Slot]
<b>Type</b> <code>dfio.data-column::non-negative-integer</code>	
<b>Initform</b> <code>0</code>	
<code>map-table</code>	[Slot]
<b>Type</b> <code>dfio.string-table:string-table</code>	
<b>Initargs</b> <code>:map-table</code>	
<code>string-count</code>	[Slot]
<b>Type</b> <code>dfio.data-column::non-negative-integer</code>	
<b>Initform</b> <code>0</code>	
<code>string-table</code>	[Slot]
<b>Type</b> <code>dfio.string-table:string-table</code>	
<b>Initform</b> <code>(dfio.string-table:string-table)</code>	

## 4.2 Internal definitions

### 4.2.1 Special variables

<code>+exponent-chars+</code>	[Special Variable]
Default exponent characters.	
<b>Package</b> <code>[dfio.decimal]</code> , page 8,	
<b>Source</b> <code>[decimal.lisp]</code> , page 3, (file)	

### 4.2.2 Functions

<code>copy-string-table</code> <i>INSTANCE</i>	[Function]
<b>Package</b> <code>[dfio.string-table]</code> , page 8,	
<b>Source</b> <code>[string-table.lisp]</code> , page 3, (file)	
<code>csv-to-data-columns</code> <i>STREAM-OR-STRING SKIP-FIRST-ROW?</i> &key <i>MAP-ALIST</i>	[Function]
Read a CSV file (or stream, or string), accumulate the values in DATA-COLUMNS, return a list of these. Rows are checked to have the same number of elements.	
When SKIP-FIRST-ROW?, the first row is read separately and returned as the second value (list of strings), otherwise it is considered data like all other rows.	
<b>Package</b> <code>[dfio]</code> , page 7,	
<b>Source</b> <code>[delimited-text.lisp]</code> , page 5, (file)	

**gobble-positive-integer** *STRING START END* [Function]

If (SUBSEQ STRING START END) starts with a nonnegative integer (ie a sequence of digits 0-9), return the integer and position at which it ends as two values.

Otherwise, return NIL and 0.

START < END has to hold, END cannot be NIL. Consequences are undefined when START >= END.

**Package** [dfio.decimal], page 8,

**Source** [decimal.lisp], page 3, (file)

**gobble-sign** *STRING START* [Function]

Return (values SIGNUM INDEX), where SIGNUM is -1 or 1 depending on whether (CHAR STRING START) was a sign, and INDEX is the index of the subsequent character (START or START+1).

**Package** [dfio.decimal], page 8,

**Source** [decimal.lisp], page 3, (file)

**string-table-get** *STRING-TABLE STRING* [Function]

Synonym for GETHASH, used internally.

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**Writer** [(setf string-table-get)], page 15, (function)

**(setf string-table-get)** *VALUE STRING-TABLE STRING* [Function]

Synonym for (SETF GETHASH), used internally, checks that STRING is a string.

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**Reader** [string-table-get], page 15, (function)

**string-table-p** *OBJECT* [Function]

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

**string-table-table** *INSTANCE* [Function]

**(setf string-table-table)** *VALUE INSTANCE* [Function]

**Package** [dfio.string-table], page 8,

**Source** [string-table.lisp], page 3, (file)

### 4.2.3 Types

**non-negative-integer** () [Type]

**Package** [dfio.data-column], page 7,

**Source** [data-column.lisp], page 4, (file)



## Appendix A Indexes

### A.1 Concepts

#### D

dfio.asd.....	3
dfio/data-column.lisp.....	4
dfio/decimal.lisp.....	3
dfio/delimited-text.lisp.....	5
dfio/pkgdcl.lisp.....	3
dfio/string-table.lisp.....	3
dfio/utils.lisp.....	4
dfio/write.lisp.....	4

#### F

File, Lisp, dfio.asd.....	3
File, Lisp, dfio/data-column.lisp.....	4
File, Lisp, dfio/decimal.lisp.....	3
File, Lisp, dfio/delimited-text.lisp.....	5
File, Lisp, dfio/pkgdcl.lisp.....	3

File, Lisp, dfio/string-table.lisp.....	3
File, Lisp, dfio/utils.lisp.....	4
File, Lisp, dfio/write.lisp.....	4

#### L

Lisp File, dfio.asd.....	3
Lisp File, dfio/data-column.lisp.....	4
Lisp File, dfio/decimal.lisp.....	3
Lisp File, dfio/delimited-text.lisp.....	5
Lisp File, dfio/pkgdcl.lisp.....	3
Lisp File, dfio/string-table.lisp.....	3
Lisp File, dfio/utils.lisp.....	4
Lisp File, dfio/write.lisp.....	4

## A.2 Functions

(	
(setf string-table-get)	15
(setf string-table-table)	15

## C

copy-string-table	14
csv-to-data-columns	14

## D

data-column	9
data-column-add	9
data-column-counts	9
data-column-vector	9

## F

Function, (setf string-table-get)	15
Function, (setf string-table-table)	15
Function, copy-string-table	14
Function, csv-to-data-columns	14
Function, data-column	9
Function, data-column-add	9
Function, data-column-counts	9
Function, data-column-vector	9
Function, gobble-positive-integer	15
Function, gobble-sign	15
Function, parse-rational	9
Function, parse-real	10
Function, read-csv	10
Function, string-table	10
Function, string-table-add	11
Function, string-table-count	11
Function, string-table-get	15
Function, string-table-intern	11
Function, string-table-lookup	11
Function, string-table-p	15
Function, string-table-strings	11
Function, string-table-table	15
Function, string-to-keyword	11

Function, string-to-symbol	11
Function, write-csv	12

## G

gobble-positive-integer	15
gobble-sign	15

## M

Macro, save	9
Macro, write-df	9

## P

parse-rational	9
parse-real	10

## R

read-csv	10
----------	----

## S

save	9
string-table	10
string-table-add	11
string-table-count	11
string-table-get	15
string-table-intern	11
string-table-lookup	11
string-table-p	15
string-table-strings	11
string-table-table	15
string-to-keyword	11
string-to-symbol	11

## W

write-csv	12
write-df	9

## A.3 Variables

**+**

`+exponent-chars+` ..... 14

**D**

`default-float-format` ..... 13

**F**

`float-count` ..... 13

**I**

`integer-count` ..... 14

`integer-max` ..... 14

`integer-min` ..... 14

**M**

`map-count` ..... 14

`map-table` ..... 14

`message` ..... 12

**R**

`reverse-elements` ..... 13

**S**

Slot, `default-float-format` ..... 13

Slot, `float-count` ..... 13

Slot, `integer-count` ..... 14

Slot, `integer-max` ..... 14

Slot, `integer-min` ..... 14

Slot, `map-count` ..... 14

Slot, `map-table` ..... 14

Slot, `message` ..... 12

Slot, `reverse-elements` ..... 13

Slot, `string` ..... 12

Slot, `string-count` ..... 14

Slot, `string-table` ..... 14

Slot, `table` ..... 13

Special Variable, `+exponent-chars+` ..... 14

`string` ..... 12

`string-count` ..... 14

`string-table` ..... 14

**T**

`table` ..... 13

## A.4 Data types

### C

Class, <code>data-column</code> .....	13
Condition, <code>parse-rational-error</code> .....	12
Condition, <code>string-table-duplicate</code> .....	12
Condition, <code>string-table-not-found</code> .....	13

### D

<code>data-column</code> .....	13
<code>dfio</code> .....	1, 7
<code>dfio.data-column</code> .....	7
<code>dfio.decimal</code> .....	8
<code>dfio.string-table</code> .....	8

### N

<code>non-negative-integer</code> .....	15
---	----

### P

Package, <code>dfio</code> .....	7
Package, <code>dfio.data-column</code> .....	7
Package, <code>dfio.decimal</code> .....	8
Package, <code>dfio.string-table</code> .....	8
<code>parse-rational-error</code> .....	12

### S

<code>string-table</code> .....	13
<code>string-table-duplicate</code> .....	12
<code>string-table-not-found</code> .....	13
Structure, <code>string-table</code> .....	13
System, <code>dfio</code> .....	1

### T

Type, <code>non-negative-integer</code> .....	15
---	----