

# The Data Frame I/O Reference Manual

---

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

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.....	5
2.1.8	dfio/delimited-text.lisp.....	5
<b>3</b>	<b>Packages .....</b>	<b>7</b>
3.1	dfio.string-table.....	7
3.2	dfio.data-column.....	7
3.3	dfio.....	8
3.4	dfio.decimal.....	8
<b>4</b>	<b>Definitions .....</b>	<b>11</b>
4.1	Exported definitions.....	11
4.1.1	Macros.....	11
4.1.2	Functions.....	11
4.1.3	Conditions.....	14
4.1.4	Structures.....	15
4.1.5	Classes.....	15
4.2	Internal definitions.....	16
4.2.1	Special variables.....	16
4.2.2	Macros.....	16
4.2.3	Functions.....	17
4.2.4	Types.....	18
<b>Appendix A</b>	<b>Indexes .....</b>	<b>19</b>
A.1	Concepts.....	19
A.2	Functions.....	20
A.3	Variables.....	21
A.4	Data types.....	22



# 1 Systems

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

## 1.1 dfio

**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
- fare-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 5, (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.string-table], page 7,
- [dfio.data-column], page 7,
- [dfio], 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 11, (function)
- [parse-rational-error], page 14, (condition)
- [parse-real], page 12, (function)

**Internal Definitions**

- [+exponent-chars+], page 16, (special variable)
- [gobble-positive-integer], page 17, (function)
- [gobble-sign], page 17, (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 12, (function)
- [string-table], page 15, (structure)
- [string-table-add], page 13, (function)
- [string-table-count], page 13, (function)

- [string-table-duplicate], page 14, (condition)
- [string-table-intern], page 13, (function)
- [string-table-lookup], page 13, (function)
- [string-table-not-found], page 14, (condition)
- [string-table-strings], page 13, (function)

#### Internal Definitions

- [copy-string-table], page 17, (function)
- [string-table-get], page 17, (function)
- [(setf string-table-get)], page 18, (function)
- [string-table-p], page 18, (function)
- [string-table-table], page 18, (function)
- [(setf string-table-table)], page 18, (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 11, (function)
- [data-column], page 15, (class)
- [data-column-add], page 11, (function)
- [data-column-counts], page 11, (function)
- [data-column-vector], page 11, (function)

#### Internal Definitions

[non-negative-integer], page 18, (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 13, (function)
- [string-to-symbol], page 13, (function)
- [symbol-name-to-pathname], page 13, (function)

#### Internal Definitions

- [%in-stream], page 17, (function)
- [%out-stream], page 17, (function)
- [\*default-external-format\*], page 16, (special variable)
- [str-strm-file], page 18, (type)
- [with-csv-input-stream], page 16, (macro)
- [with-csv-output-stream], page 16, (macro)

### 2.1.7 dfio/write.lisp

**Dependency**

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

**Parent**

[dfio], page 1, (system)

**Location**

write.lisp

**Exported Definitions**

- [save], page 11, (macro)
- [write-df], page 11, (macro)

### 2.1.8 dfio/delimited-text.lisp

**Dependency**

[write.lisp], page 5, (file)

**Parent**

[dfio], page 1, (system)

**Location**

delimited-text.lisp

**Exported Definitions**

- [read-csv], page 12, (function)
- [write-csv], page 14, (function)

**Internal Definitions**

[csv-to-data-columns], page 17, (function)





## 3 Packages

Packages are listed by definition order.

### 3.1 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 12, (function)
- [string-table], page 15, (structure)
- [string-table-add], page 13, (function)
- [string-table-count], page 13, (function)
- [string-table-duplicate], page 14, (condition)
- [string-table-intern], page 13, (function)
- [string-table-lookup], page 13, (function)
- [string-table-not-found], page 14, (condition)
- [string-table-strings], page 13, (function)

**Internal Definitions**

- [copy-string-table], page 17, (function)
- [string-table-get], page 17, (function)
- [(setf string-table-get)], page 18, (function)
- [string-table-p], page 18, (function)
- [string-table-table], page 18, (function)
- [(setf string-table-table)], page 18, (function)

### 3.2 dfio.data-column

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

**Use List**

- let-plus
- [dfio.string-table], page 7,
- [dfio.decimal], page 8,
- anaphora
- common-lisp

**Used By List**

[dfio], page 8,

**Exported Definitions**

- [data-column], page 11, (function)

- [data-column], page 15, (class)
- [data-column-add], page 11, (function)
- [data-column-counts], page 11, (function)
- [data-column-vector], page 11, (function)

#### Internal Definitions

[non-negative-integer], page 18, (type)

### 3.3 dfio

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

#### Use List

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

#### Used By List

lisp-stat

#### Exported Definitions

- [read-csv], page 12, (function)
- [save], page 11, (macro)
- [string-to-keyword], page 13, (function)
- [string-to-symbol], page 13, (function)
- [symbol-name-to-pathname], page 13, (function)
- [write-csv], page 14, (function)
- [write-df], page 11, (macro)

#### Internal Definitions

- [%in-stream], page 17, (function)
- [%out-stream], page 17, (function)
- [\*default-external-format\*], page 16, (special variable)
- [csv-to-data-columns], page 17, (function)
- [str-strm-file], page 18, (type)
- [with-csv-input-stream], page 16, (macro)
- [with-csv-output-stream], page 16, (macro)

### 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 11, (function)
- `[parse-rational-error]`, page 14, (condition)
- `[parse-real]`, page 12, (function)

**Internal Definitions**

- `[+exponent-chars+]`, page 16, (special variable)
- `[gobble-positive-integer]`, page 17, (function)
- `[gobble-sign]`, page 17, (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 8,

**Source** [write.lisp], page 5, (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 8,

**Source** [write.lisp], page 5, (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 8,

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

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

**Package** [dfio.string-table], page 7,

**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 7,

**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 7,

**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 7,

**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 7,

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

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

**Package** [dfio.string-table], page 7,

**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 8,

**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 8,

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

**symbol-name-to-pathname** *STRING* [Function]  
 Map the symbol-name of *S* to something that can be part of a logical-pathname

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

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



**write-csv** *DF* *STREAM* &key *ADD-FIRST-ROW* (*SEPARATOR* *SEPARATOR*) (*QUOTE* *QUOTE*) (*EOL* *EOL*) [Function]

Write *DF* to *STRING-OR-STREAM* in CSV format. *STRING-OR-STREAM* can be a *STREAM*, a *STRING* or a file *PATHSPEC*.

Keywords:

*string-or-stream*: stream to write to. Default: nil, returning a string

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

*separator*: separator to use when reading or writing CSV files. A character. By default, a comma: #,

*quote*: quote character to use when reading or writing CSV files. A character. By default, a double-quote: #" *eol*: line ending to use when writing CSV files. A string. By default, +CRLF+ as specified by *creativyst*.

Notes:

The *:newline* keyword requires a sequence, so use *:newline* '(#newline)

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

**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 7,

**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 7,

**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 7,

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

**Direct superclasses**

structure-object (structure)

**Direct methods**

print-object (method)

**Direct slots**

table [Slot]

**Type** hash-table

**Initform** (make-hash-table :test (function equalp))

**Readers** [string-table-table], page 18, (function)

**Writers** [(setf string-table-table)], page 18, (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**

reverse-elements [Slot]

**Type** list

default-float-format [Slot]

**Type** symbol

**Initargs** :default-float-format

float-count [Slot]

**Type** dfio.data-column::non-negative-integer

**Initform** 0

integer-count [Slot]

**Type** dfio.data-column::non-negative-integer

**Initform** 0

integer-min [Slot]

**Type** integer

**Initform** 0

<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

<b>*default-external-format*</b>		[Special Variable]
External format used for opening files		
<b>Package</b>	<code>[dfio]</code> , page 8,	
<b>Source</b>	<code>[utils.lisp]</code> , page 4, (file)	
<b>+exponent-chars+</b>		[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 Macros

<code>with-csv-input-stream</code> ( <i>NAME INP</i> ) <b>&amp;body</b> <i>BODY</i>		[Macro]
<b>Package</b>	<code>[dfio]</code> , page 8,	
<b>Source</b>	<code>[utils.lisp]</code> , page 4, (file)	
<code>with-csv-output-stream</code> ( <i>NAME INP</i> ) <b>&amp;body</b> <i>BODY</i>		[Macro]
<b>Package</b>	<code>[dfio]</code> , page 8,	
<b>Source</b>	<code>[utils.lisp]</code> , page 4, (file)	

### 4.2.3 Functions

**%in-stream** *STREAM-OR-STRING* [Function]

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

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

**%out-stream** *STREAM-OR-STRING* [Function]

creates a stream from the given thing, trying to DWIM

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

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

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

**Package** [dfio.string-table], page 7,

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

**csv-to-data-columns** *STREAM-OR-STRING SKIP-FIRST-ROW?* [Function]  
**&key** *MAP-ALIST*

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.

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

**Source** [delimited-text.lisp], 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 7,

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

**Writer** [(setf string-table-get)], page 18, (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 7,  
**Source** [string-table.lisp], page 3, (file)  
**Reader** [string-table-get], page 17, (function)

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

**Package** [dfio.string-table], page 7,  
**Source** [string-table.lisp], page 3, (file)

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

**Package** [dfio.string-table], page 7,  
**Source** [string-table.lisp], page 3, (file)

#### 4.2.4 Types

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

**Package** [dfio.data-column], page 7,  
**Source** [data-column.lisp], page 4, (file)

**str-strm-file ()** [Type]

**Package** [dfio], page 8,  
**Source** [utils.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.....	5

#### 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.....	5

#### 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.....	5

## A.2 Functions

### %

%in-stream .....	17
%out-stream .....	17

### (

(setf string-table-get) .....	18
(setf string-table-table) .....	18

### C

copy-string-table .....	17
csv-to-data-columns .....	17

### D

data-column .....	11
data-column-add .....	11
data-column-counts .....	11
data-column-vector .....	11

### F

Function, %in-stream .....	17
Function, %out-stream .....	17
Function, (setf string-table-get) .....	18
Function, (setf string-table-table) .....	18
Function, copy-string-table .....	17
Function, csv-to-data-columns .....	17
Function, data-column .....	11
Function, data-column-add .....	11
Function, data-column-counts .....	11
Function, data-column-vector .....	11
Function, gobble-positive-integer .....	17
Function, gobble-sign .....	17
Function, parse-rational .....	11
Function, parse-real .....	12
Function, read-csv .....	12
Function, string-table .....	12
Function, string-table-add .....	13
Function, string-table-count .....	13
Function, string-table-get .....	17
Function, string-table-intern .....	13
Function, string-table-lookup .....	13
Function, string-table-p .....	18
Function, string-table-strings .....	13
Function, string-table-table .....	18
Function, string-to-keyword .....	13

Function, string-to-symbol .....	13
Function, symbol-name-to-pathname .....	13
Function, write-csv .....	14

### G

gobble-positive-integer .....	17
gobble-sign .....	17

### M

Macro, save .....	11
Macro, with-csv-input-stream .....	16
Macro, with-csv-output-stream .....	16
Macro, write-df .....	11

### P

parse-rational .....	11
parse-real .....	12

### R

read-csv .....	12
----------------	----

### S

save .....	11
string-table .....	12
string-table-add .....	13
string-table-count .....	13
string-table-get .....	17
string-table-intern .....	13
string-table-lookup .....	13
string-table-p .....	18
string-table-strings .....	13
string-table-table .....	18
string-to-keyword .....	13
string-to-symbol .....	13
symbol-name-to-pathname .....	13

### W

with-csv-input-stream .....	16
with-csv-output-stream .....	16
write-csv .....	14
write-df .....	11

## A.3 Variables

### \*

`*default-external-format*` ..... 16

### +

`+exponent-chars+` ..... 16

### D

`default-float-format` ..... 15

### F

`float-count` ..... 15

### I

`integer-count` ..... 15

`integer-max` ..... 16

`integer-min` ..... 15

### M

`map-count` ..... 16

`map-table` ..... 16

`message` ..... 14

### R

`reverse-elements` ..... 15

### S

Slot, `default-float-format` ..... 15

Slot, `float-count` ..... 15

Slot, `integer-count` ..... 15

Slot, `integer-max` ..... 16

Slot, `integer-min` ..... 15

Slot, `map-count` ..... 16

Slot, `map-table` ..... 16

Slot, `message` ..... 14

Slot, `reverse-elements` ..... 15

Slot, `string` ..... 14

Slot, `string-count` ..... 16

Slot, `string-table` ..... 16

Slot, `table` ..... 15

Special Variable, `*default-external-format*` ..... 16

Special Variable, `+exponent-chars+` ..... 16

`string` ..... 14

`string-count` ..... 16

`string-table` ..... 16

### T

`table` ..... 15



## A.4 Data types

### C

Class, <code>data-column</code> .....	15
Condition, <code>parse-rational-error</code> .....	14
Condition, <code>string-table-duplicate</code> .....	14
Condition, <code>string-table-not-found</code> .....	14

### D

<code>data-column</code> .....	15
<code>dfio</code> .....	1, 8
<code>dfio.data-column</code> .....	7
<code>dfio.decimal</code> .....	8
<code>dfio.string-table</code> .....	7

### N

<code>non-negative-integer</code> .....	18
---	----

### P

Package, <code>dfio</code> .....	8
Package, <code>dfio.data-column</code> .....	7
Package, <code>dfio.decimal</code> .....	8
Package, <code>dfio.string-table</code> .....	7
<code>parse-rational-error</code> .....	14

### S

<code>str-strm-file</code> .....	18
<code>string-table</code> .....	15
<code>string-table-duplicate</code> .....	14
<code>string-table-not-found</code> .....	14
Structure, <code>string-table</code> .....	15
System, <code>dfio</code> .....	1

### T

Type, <code>non-negative-integer</code> .....	18
Type, <code>str-strm-file</code> .....	18