

Package: swat (via r-universe)

May 27, 2026

Type Package

Title SAS Scripting Wrapper for Analytics Transfer (SWAT)

Version 1.10.0.9000

TKVersion none

Date 06JUN2020

Author Jared Dean [aut, cre], Tom Weber [aut, cre], Kevin Smith [aut]

Maintainer Kevin Smith <Kevin.Smith@sas.com>

Description SWAT provides an R scripting interface to SAS Cloud Analytic Services (CAS). This enables R programmers to perform analytical functions provided by SAS.

License file LICENSE

Copyright Copyright 2017 SAS Institute Inc. All Rights Reserved.

Issues Enter issues at <https://github.com/sassoftware/r-swat/issues>.

Depends R (>= 3.1.0)

Suggests testthat, xlsx, knitr

Imports httr, jsonlite, tools, methods

Collate 'datetime.R' 'rswat.R' 'authinfo.R' 'rswat_rest.R' 'swat.R'
'helper.R' 'CASTab.R' 'table_functions.R' 'descriptive_stats.R'
'action.R' 'CASdf.R' 'read_write.R' 'analytical_functions.R'
'graphics.R' 'Ops.R' 'connection_viewer.R'

RoxygenNote 7.3.2

VignetteBuilder knitr

Config/pak/sysreqs libssl-dev

Repository <https://pinduzera.r-universe.dev>

Date/Publication 2025-10-29 18:02:29 UTC

RemoteUrl <https://github.com/pinduzera/R-swat-universe>

RemoteRef HEAD

RemoteSha 6b0488c48cc57f319cc7e84457317defb3f52b24

Contents

[,CASTable-method	3
as.casDataFrame	7
as.casTable	8
CAS Actions	9
CAS-class	10
cas.close	11
cas.count	11
cas.css	12
cas.cv	13
cas.max	14
cas.mean	15
cas.median	16
cas.min	17
cas.mode	18
cas.nmiss	18
cas.probt	19
cas.quantile	20
cas.read.csv	21
cas.read.jmp	22
cas.read.sas7bdat	24
cas.read.table	25
cas.read.xlsx	28
cas.readRDS	30
cas.saveRDS	31
cas.sd	32
cas.stderr	33
cas.sum	34
cas.terminate	35
cas.tvalue	35
cas.upload	36
cas.upload.file	36
cas.upload.frame	37
cas.uss	37
cas.var	38
cas.write.csv	39
cas.write.csv2	40
cas.write.table	41
cas.write.xlsx	42
casDataFrame-class	43
CASTable-class	44
cbind.casTable	45
cbind2.casTable	45
colMeans,CASTable-method	46
colnames,CASTable-method	47
colSums,CASTable-method	48
cor,CASTable-method	48

cov,CASTable-method	49
defCasTable	50
dim,CASTable-method	51
dimnames,CASTable-method	52
download_sas_binaries	52
dropTable	53
head,CASTable-method	54
is.castable	54
length,CASTable-method	55
listActionParms	56
listActionSets	56
loadActionSet	57
max,CASTable-method	58
mean,CASTable-method	58
median,CASTable-method	59
min,CASTable-method	60
names,CASTable-method	60
ncol,CASTable-method	61
nrow,CASTable-method	62
rbind,CASTable-method	62
rbind.casTable	63
rbind2.casTable	63
rownames,CASTable-method	64
runAction	64
show,casDataFrame-method	65
subset.casTable	66
summary,CASTable-method	66
swat	67
tail,CASTable-method	68
to.casDataFrame	69
to.data.frame	70
to.r.data.frame	70
unique,CASTable-method	71
Index	72

[,CASTable-method	<i>Extract Columns from a CAS Table</i>
-------------------	---

Description

Extract Columns from a CAS Table

Usage

```
## S4 method for signature 'CASTable'
x[i, j, ..., drop = TRUE]

## S4 replacement method for signature 'CASTable'
x[i, j, ...] <- value

## S4 method for signature 'CASTable'
x[[i]]

## S4 method for signature 'CASTable'
x$name

## S4 replacement method for signature 'CASTable'
x$name <- value

.cas.arith(e1, op, e2)

## S4 method for signature 'CASTable,CASTable'
e1 - e2

## S4 method for signature 'CASTable,ANY'
e1 - e2

## S4 method for signature 'ANY,CASTable'
e1 - e2

## S4 method for signature 'CASTable,CASTable'
e1 + e2

## S4 method for signature 'CASTable,ANY'
e1 + e2

## S4 method for signature 'ANY,CASTable'
e1 + e2

## S4 method for signature 'CASTable,CASTable'
e1 / e2

## S4 method for signature 'CASTable,ANY'
e1 / e2

## S4 method for signature 'ANY,CASTable'
e1 / e2

## S4 method for signature 'CASTable,CASTable'
e1 * e2
```

```
## S4 method for signature 'CASTable,ANY'
e1 * e2

## S4 method for signature 'ANY,CASTable'
e1 * e2

## S4 method for signature 'CASTable,CASTable'
e1 ^ e2

## S4 method for signature 'CASTable,ANY'
e1 ^ e2

## S4 method for signature 'ANY,CASTable'
e1 ^ e2

## S4 method for signature 'CASTable,CASTable'
e1 %% e2

## S4 method for signature 'CASTable,ANY'
e1 %% e2

## S4 method for signature 'ANY,CASTable'
e1 %% e2

## S4 method for signature 'CASTable,CASTable'
e1 %/% e2

## S4 method for signature 'CASTable,ANY'
e1 %/% e2

## S4 method for signature 'ANY,CASTable'
e1 %/% e2

.cas.compare(e1, op, e2)

## S4 method for signature 'CASTable,CASTable'
e1 > e2

## S4 method for signature 'CASTable,ANY'
e1 > e2

## S4 method for signature 'ANY,CASTable'
e1 > e2

## S4 method for signature 'CASTable,CASTable'
e1 < e2

## S4 method for signature 'CASTable,ANY'
```

```
e1 < e2

## S4 method for signature 'ANY,CASTable'
e1 < e2

## S4 method for signature 'CASTable,CASTable'
e1 >= e2

## S4 method for signature 'CASTable,ANY'
e1 >= e2

## S4 method for signature 'ANY,CASTable'
e1 >= e2

## S4 method for signature 'CASTable,CASTable'
e1 <= e2

## S4 method for signature 'CASTable,ANY'
e1 <= e2

## S4 method for signature 'ANY,CASTable'
e1 <= e2

## S4 method for signature 'CASTable,CASTable'
e1 == e2

## S4 method for signature 'CASTable,ANY'
e1 == e2

## S4 method for signature 'ANY,CASTable'
e1 == e2

## S4 method for signature 'CASTable,CASTable'
e1 != e2

## S4 method for signature 'CASTable,ANY'
e1 != e2

## S4 method for signature 'ANY,CASTable'
e1 != e2

.cas.logic(e1, op, e2)

## S4 method for signature 'CASTable,CASTable'
e1 & e2

## S4 method for signature 'CASTable,ANY'
e1 & e2
```

```
## S4 method for signature 'ANY,CASTable'  
e1 & e2  
  
## S4 method for signature 'CASTable,CASTable'  
e1 | e2  
  
## S4 method for signature 'CASTable,ANY'  
e1 | e2  
  
## S4 method for signature 'ANY,CASTable'  
e1 | e2  
  
## S4 method for signature 'CASTable'  
!x
```

as.casDataFrame

Convert an R Data Frame to a CAS Data Frame

Description

This function is rarely used for programming. It is used by the package to associate CAS metadata with tabular data that is returned by CAS actions.

Usage

```
as.casDataFrame(  
  df,  
  name = "",  
  label = "",  
  title = "",  
  attrs = list(),  
  col.labels = "",  
  col.formats = "",  
  col.attrs = list(),  
  col.sizes = list(),  
  col.types = "",  
  col.widths = 0  
)
```

Value

casDataFrame

Examples

```
## Not run:
cdf2 = as.casDataFrame(df3[1:4])
cdf = as.casDataFrame(iris)

## End(Not run)
```

as.casTable

Upload an Object to a CAS Table

Description

Uploads an R data frame to CAS and returns a [CASTable](#) object. The CASTable object is a reference in R (the client) to the in-memory table that is in CAS (the server).

Usage

```
as.casTable(conn, df, casOut = "")
```

Arguments

conn	A CAS object that represents a connection and session in CAS.
df	A data.frame object with the data to upload to CAS.
casOut	An optional character or list. If you specify a string, then the string is used as the in-memory table name. A list can be used to specify properties for the in-memory table as follows: <ul style="list-style-type: none"> name An optional character that specifies the name for the in-memory table. By default, the name of the data frame is used. caslib An optional character that specifies the caslib. Specify this parameter to override the active caslib. label An optional character that specifies a descriptive label for the data. replace An optional logical. When set to TRUE, you can replace an existing in-memory table with the same name in the same caslib. The default value is FALSE. promote An optional logical. When set to TRUE, the in-memory table has global scope and can be available to other CAS sessions (subject to access controls). The default value is FALSE and the in-memory table has session scope so that it is accessible with the session that uploaded the table only. Session-scope tables are ideal for data analysis. Global-scope tables are better suited for reporting. replication An optional numeric that specifies the number of redundant copies of in-memory blocks. This parameter applies to distributed servers only. The default value is 1.

Value

[CASTable](#)

Examples

```
## Not run:
s      <- CAS('cloud.example.com', 5570)
irisct <- as.casTable(s, iris)

# Specify a name for the in-memory table.
mtcarsct <- as.casTable(s, mtcars, casOut="mtcarsct")

# Avoid replacing an existing in-memory table.
mtcarsct <- as.casTable(s, mtcars, casOut=list(name="mtcarsct", replace=FALSE))

## End(Not run)
```

CAS Actions

Common Page for CAS Actions

Description

When you connect to SAS Cloud Analytic Services (CAS), the SWAT package software generates an R function for each CAS action that is available on the server.

Usage

```
cas.actionSet.action(CASorCASTab, parameters...)
```

Arguments

CASorCASTab	An instance of a CAS object that represents a connection and CAS session, or an instance of a CASTable .
parameters	Actions accept a series of parameters in key=value pair format. The parameters are action-specific. See the product documentation.

Examples

The following two functions are generated and correspond to the `table.tableInfo` and `simple.summary` actions:

```
cas.table.tableInfo(irisct)
cas.simple.summary(irisct)
```

Product Documentation

For a list of all the CAS actions that are available with SAS Visual Analytics, SAS Visual Statistics, and SAS Visual Data Mining and Machine Learning, see the following URL:

[Actions and Action Sets by Name and Product](#)

For the latest product documentation for SAS Viya, see [Documentation for SAS Viya](#).

CAS-class

CAS Object Class

Description

An instance of this class represents a connection and session between the client (R) and the server (SAS Cloud Analytic Services).

Arguments

hostname	A character string that specifies the host name of the CAS controller.
port	A numeric value that specifies the network port number that the CAS controller listens on.
protocol	A string that specifies one of the following: cas use binary communication. This is the default. http use HTTP communication with the REST interface on the CAS controller. https use HTTPS communication with the REST interface on the CAS controller. This protocol must be specified explicitly. auto automatically detect between the binary and HTTP.
username	A character string that identifies the user ID to authenticate as.
password	A character string that specifies your password.
session	A character string that identifies the 32 character UUID of an existing session, if you want to connect to an existing session. This is rare.
locale	A character string that specifies the locale to use for the CAS session. By default, the locale is set to the locale of the server.
authinfo	A character string that specifies an alternative path to a .authinfo file that is used for authentication. By default, ~/.authinfo is used on Linux and %HOMEDRIVE%\%HOMEPATH%_authinfo is used on Windows.
path	Base path of the connection URL
authcode	Authorization code from SASLogon used to retrieve an OAuth token.

Value

A CAS object.

Examples

```
## Not run:
# Use binary communication and credentials from the default authinfo location.
s <- CAS('cloud.example.com', 5570)

# Use HTTPS and credentials from the default authinfo location.
s <- CAS('cloud.example.com', 8777, protocol='https')

# Use binary or HTTP communication and credentials from an alternative authinfo.
s <- CAS('cloud.example.com', 5570, protocol='auto', authinfo=~/.alt.txt)

# Use binary or HTTP communication and specify credentials.
s <- CAS('cloud.example.com', 8777, protocol='auto', username="sasdemo"
  password="!s3cret")

## End(Not run)
```

cas.close

Close a CAS connection while leaving the session alive

Description

Close a CAS connection while leaving the session alive

Usage

```
cas.close(conn)
```

Arguments

CAS The CAS connection object

cas.count

Count of Nonmissing Values

Description

Returns the number of nonmissing values in the input table by column.

Usage

```
cas.count(CASTable)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

casDataFrame

The result includes one row for each numeric variable and a column that is named N for the non-missing count.

See Also

cas.nmiss to count missing values.

Examples

```
## Not run:  
cas.count(ct[1:4])  
cas.count(ct$n2)  
  
## End(Not run)
```

cas.css

Corrected Sum of Squares

Description

Returns the corrected sum of squares of the values for each column in the input table.

Usage

```
cas.css(CASTable)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

vector

The result is a named numeric vector. You can access the corrected sum of squares by column name or index.

Examples

```
## Not run:  
x <- cas.css(irisct)  
x['Sepal.Length']  
x[1:2]  
  
## End(Not run)
```

cas.cv

Coefficient of Variation

Description

Returns the coefficient of variation of the values for each column in the input table.

Usage

```
cas.cv(CASTable)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

vector

The result is a named numeric vector. You can access the coefficient of variation by column name or index.

Examples

```
## Not run:  
x <- cas.cv(irisct)  
x['Sepal.Length']  
x[1:2]  
  
## End(Not run)
```

cas.max	<i>Maximum Values</i>
---------	-----------------------

Description

Returns the maximum value for each column in the input table.

Usage

```
cas.max(CASTable)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

casDataFrame

The result includes one row for each numeric variable and a column that is named Max for the maximum value.

See Also

max, CASTable-method

Examples

```
## Not run:  
cas.max(ct[1:4])  
cas.max(ct$n2)  
  
## End(Not run)
```

cas.mean	<i>Average Values</i>
----------	-----------------------

Description

Returns the mean value for each column in the input table.

Usage

```
cas.mean(CAStable)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

casDataFrame

The result includes one row for each numeric variable and a column that is named Mean for the mean value.

See Also

mean, CAStable-method

Examples

```
## Not run:  
cas.mean(ct[1:4])  
cas.mean(ct$n2)  
  
## End(Not run)
```

cas.median	<i>Median Values</i>
------------	----------------------

Description

Returns the median value for each column in the input table.

Usage

```
cas.median(CASTable, q)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

data.frame

The result includes one row for each numeric variable and a column that is named Median for the median value.

See Also

median,CASTable-method

Examples

```
## Not run:  
cas.median(ct[1:4])  
cas.median(ct$n2)  
  
## End(Not run)
```

cas.min	<i>Minimum Values</i>
---------	-----------------------

Description

Returns the minimum value for each column in the input table.

Usage

```
cas.min(CASTable)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

casDataFrame

The result includes one row for each numeric variable and a column that is named Minimum for the minimum value.

See Also

min, CASTable-method

Examples

```
## Not run:  
cas.min(ct[1:4])  
cas.min(ct$n2)  
  
## End(Not run)
```

cas.mode

Mode Value

Description

Returns the value that occurs most often for each column in the input table and the count for the value.

Usage

```
cas.mode(CASTable)
```

Arguments

x CASTable.

Details

This function operates on numeric and character columns.

Value

data.frame

The result includes one row for each variable. One column is named Mode for the most common value. Another column is named Count to show the number of rows with the most common value.

Examples

```
## Not run:  
cas.mode(ct[1:4])  
cas.mode(ct$n2)  
  
## End(Not run)
```

cas.nmiss*Number of Missing Values*

Description

Returns the number of missing values in the input table by column.

Usage

```
cas.nmiss(CASTable)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

vector

The result is a named numeric vector. You can access the count of missing values by column name or index.

See Also

cas.count to count nonmissing values.

Examples

```
## Not run:  
x <- cas.nmiss(irisct)  
x['Sepal.Length']  
x[1:2]  
  
## End(Not run)
```

cas.probt

P-Value of the T-Statistics

Description

Returns the p-values for the values for each column in the input table.

Usage

```
cas.probt(CASTable)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

vector

The result is a named numeric vector. You can access the p-value by column name or index.

Examples

```
## Not run:  
x <- cas.probt(irisct)  
x['Sepal.Length']  
x[1:2]  
  
## End(Not run)
```

`cas.quantile`*Quantile and Percentile Values*

Description

Returns the requested percentiles for each column in the input table.

Usage

```
cas.quantile(CASTable, q)
```

Arguments

q	A list of numeric values.
x	CASTable.

Details

This function operates on numeric columns only.

Value

data.frame

The result includes one row for the variable, the requested percentile, and the value.

Examples

```
## Not run:  
cas.quantile(ct[1:4], q=50)  
cas.quantile(ct$n2, q=c(10, 25, 50, 75, 90))  
  
## End(Not run)
```

cas.read.csv	<i>Read a CSV File and Upload to a CAS Table</i>
--------------	--

Description

This function is a convenience wrapper for the R `read.csv` and `as.casTable` functions. After reading the file that is accessible to the R client, it is uploaded to an in-memory table in CAS (the server).

Usage

```
cas.read.csv(
  conn,
  file,
  header = TRUE,
  sep = ",",
  quote = "\"",
  dec = ".",
  fill = TRUE,
  comment.char = "",
  casOut = list(name = "", replace = FALSE),
  ...
)
```

Arguments

<code>conn</code>	An instance of a CAS object that represents a connection and CAS session.		
<code>file</code>	An character string that specifies the filename or connection for the data to read.		
<code>header</code>	An optional logical that specifies whether the first line of the file contains variable names.		
<code>sep</code>	A character that specifies the field delimiter. This value is passed to <code>read.csv</code> .		
<code>quote</code>	A character string that specifies the characters that enclose character data type variables. This value is passed to <code>read.csv</code> .		
<code>dec</code>	An optional character to represent the decimal separator. This value is passed to <code>read.csv</code> .		
<code>fill</code>	An optional logical value. When set to <code>TRUE</code> , blank fields are implicitly added for rows that have unequal length. This value is passed to <code>read.csv</code> .		
<code>comment.char</code>	An optional character that specifies the character to interpret as the beginning of a comment. This value is passed to <code>read.csv</code> .		
<code>casOut</code>	An optional character or list. If you specify a string, then the string is used as the in-memory table name. A list can be used to specify properties for the in-memory table as follows: <table style="margin-left: 20px; border-collapse: collapse;"> <tr> <td style="width: 15%;"><code>name</code></td> <td>An optional character that specifies the name for the in-memory table. By default, the name of the data frame is used.</td> </tr> </table>	<code>name</code>	An optional character that specifies the name for the in-memory table. By default, the name of the data frame is used.
<code>name</code>	An optional character that specifies the name for the in-memory table. By default, the name of the data frame is used.		

caslib An optional character that specifies the caslib. Specify this parameter to override the active caslib.

label An optional character that specifies a descriptive label for the data.

replace An optional logical. When set to TRUE, you can replace an existing in-memory table with the same name in the same caslib. The default value is FALSE.

promote An optional logical. When set to TRUE, the in-memory table has global scope and can be available to other CAS sessions (subject to access controls). The default value is FALSE and the in-memory table has session scope so that it is accessible with the session that uploaded the table only. Session-scope tables are ideal for data analysis. Global-scope tables are better suited for reporting.

replication An optional numeric that specifies the number of redundant copies of in-memory blocks. This parameter applies to distributed servers only. The default value is 1.

... Optional parameters that are passed to read.csv.

Value

CASTable

See Also

Other functions for loading in-memory data: [cas.read.jmp\(\)](#), [cas.read.sas7bdat\(\)](#), [cas.read.table\(\)](#), [cas.read.xlsx\(\)](#), [cas.readRDS\(\)](#)

Examples

```
## Not run:
# Upload a CSV, the in-memory table is named HEART
heartct <- cas.read.csv(s, "http://support.sas.com/documentation/
  onlinedoc/viya/exampledatasets/heart.csv")

# Upload the same CSV, name the in-memory table HEARTCT
heartct <- cas.read.csv(s, "http://support.sas.com/documentation/
  onlinedoc/viya/exampledatasets/heart.csv",
  casOut=list(name="heartct", replace=TRUE))

## End(Not run)
```

cas.read.jmp

Upload a JMP File to a CAS Table

Description

This function transfers a JMP file (.jmp) from R (the client) to the CAS server. The server imports the data and R returns a CASTable object to reference the in-memory table in CAS.

Usage

```
cas.read.jmp(conn, file, casOut = NULL)
```

Arguments

conn	An instance of a CAS object that represents a connection and CAS session.
file	An character string that specifies the JMP file.
casOut	An optional character or list. If you specify a string, then the string is used as the in-memory table name. A list can be used to specify properties for the in-memory table as follows: <ul style="list-style-type: none"> name An optional character that specifies the name for the in-memory table. By default, the name of the data frame is used. caslib An optional character that specifies the caslib. Specify this parameter to override the active caslib. label An optional character that specifies a descriptive label for the data. replace An optional logical. When set to TRUE, you can replace an existing in-memory table with the same name in the same caslib. The default value is FALSE. promote An optional logical. When set to TRUE, the in-memory table has global scope and can be available to other CAS sessions (subject to access controls). The default value is FALSE and the in-memory table has session scope so that it is accessible with the session that uploaded the table only. Session-scope tables are ideal for data analysis. Global-scope tables are better suited for reporting. replication An optional numeric that specifies the number of redundant copies of in-memory blocks. This parameter applies to distributed servers only. The default value is 1.

Value

[CASTable](#)

See Also

Other functions for loading in-memory data: [cas.read.csv\(\)](#), [cas.read.sas7bdat\(\)](#), [cas.read.table\(\)](#), [cas.read.xlsx\(\)](#), [cas.readRDS\(\)](#)

Examples

```
## Not run:
spring_example <- cas.read.jmp(s, "/path/to/Spring\ Example.jmp",
                             casOut=list(name="spring_example"))

## End(Not run)
```

cas.read.sas7bdat *Upload a SAS Data Set to a CAS Table*

Description

This function transfers a SAS data set (.sas7bdat) file from R (the client) to the CAS server. The server imports the data and R returns a CASTable object to reference the in-memory table in CAS.

Usage

```
cas.read.sas7bdat(conn, file, casOut = list(name = "", replace = FALSE))
```

Arguments

conn	An instance of a CAS object that represents a connection and CAS session.
file	An character string that specifies the SAS data set (.sas7bdat file).
casOut	An optional character or list. If you specify a string, then the string is used as the in-memory table name. A list can be used to specify properties for the in-memory table as follows: <ul style="list-style-type: none"> name An optional character that specifies the name for the in-memory table. By default, the name of the data frame is used. caslib An optional character that specifies the caslib. Specify this parameter to override the active caslib. label An optional character that specifies a descriptive label for the data. replace An optional logical. When set to TRUE, you can replace an existing in-memory table with the same name in the same caslib. The default value is FALSE. promote An optional logical. When set to TRUE, the in-memory table has global scope and can be available to other CAS sessions (subject to access controls). The default value is FALSE and the in-memory table has session scope so that it is accessible with the session that uploaded the table only. Session-scope tables are ideal for data analysis. Global-scope tables are better suited for reporting. replication An optional numeric that specifies the number of redundant copies of in-memory blocks. This parameter applies to distributed servers only. The default value is 1.

Value

CASTable

See Also

Other functions for loading in-memory data: [cas.read.csv\(\)](#), [cas.read.jmp\(\)](#), [cas.read.table\(\)](#), [cas.read.xlsx\(\)](#), [cas.readRDS\(\)](#)

Examples

```
## Not run:
gold_medals <- cas.read.sas7bdat(s, "/path/to/gold_medals.sas7bdat")

## End(Not run)
```

cas.read.table	<i>Read a File and Upload to a CAS Table</i>
----------------	--

Description

This function is a convenience wrapper for the R `read.table` and `as.casTable` functions. After reading the file that is accessible to the R client, it is uploaded to an in-memory table in CAS (the server).

Usage

```
cas.read.table(
  conn,
  file,
  header = FALSE,
  sep = "",
  quote = "\"",
  dec = ".",
  numerals = c("allow.loss", "warn.loss", "no.loss"),
  row.names,
  col.names,
  as.is = !stringsAsFactors,
  na.strings = "NA",
  colClasses = NA,
  nrows = -1,
  skip = 0,
  check.names = TRUE,
  fill = !blank.lines.skip,
  strip.white = FALSE,
  blank.lines.skip = TRUE,
  comment.char = "#",
  allowEscapes = FALSE,
  flush = FALSE,
  stringsAsFactors = default.stringsAsFactors(),
  fileEncoding = "",
  encoding = "unknown",
  text,
  skipNul = FALSE,
  casOut = list(name = "", replace = FALSE)
)
```

Arguments

conn	An instance of a CAS object that represents a connection and CAS session.
file	An character string that specifies the filename. This value is passed to read.table.
header	An optional logical that specifies whether the first line of the file contains variable names.
sep	An optional character that is used to specify the field delimiter for the file. This value is passed to write.table.
quote	An optional character string that specifies the characters that enclose character data. The value is passed to read.table.
dec	An optional character that specifies the decimal separator. This value is passed to read.table.
numerals	An optional character string that specifies how to interpret numbers that can lose precision due to the conversion from text to double precision. This value is passed to read.table.
row.names	An optional character vector of row names. This value is passed to read.table.
col.names	An optional character vector of names for the variables. The default is to use "V" followed by the column number. This value is passed to read.table.
as.is	An optional vector of logical, numeric, or character indices that specify the columns that are not converted to factors. This value is passed to read.table.
na.strings	An optional character vector that specifies the characters to interpret as NA. This value is passed to read.table.
colClasses	An optional character vector that specifies the classes for the columns. This value is passed to read.table.
nrows	An optional numeric that specifies the maximum number of rows to read. This value is passed to read.table.
skip	An optional numeric that specifies the number of lines to skip in the file before reading data. This value is passed to read.table.
check.names	An optional logical that specifies whether variable names from the file are valid names. This value is passed to read.table.
fill	An optional logical value. When set to TRUE, blank fields are implicitly added for rows that have unequal length. This value is passed to read.table.
strip.white	An optional logical that specifies whether white space characters are stripped from character data that are not enclosed with quotation marks. This value is ignored unless sep is specified. This value is passed to read.table.
blank.lines.skip	An optional logical that specifies whether blank lines in the file are ignored. This value is passed to read.table.
comment.char	An optional character that specifies the character to interpret as the beginning of a comment. This value is passed to read.table.
allowEscapes	An optional logical that specifies whether C-style escape characters such as \n are interpreted or are read verbatim. This value is passed to read.table.

flush	An optional logical that specifies whether input that follows the last field to read is flushed. Setting this argument to TRUE enables adding comments to the end of data lines. This value is passed to read.table.
stringsAsFactors	An optional logical that specifies whether character vectors are converted to factors. This argument is overridden by as.is and colClasses. This value is passed to read.table.
fileEncoding	An optional character string that specifies the encoding to use for reading the file. This value is passed to read.table.
encoding	An optional character string that specifies the encoding for character data. This value is passed to read.table.
text	An optional character string. This value is passed to read.table.
skipNul	An optional logical that is passed to read.table.
casOut	An optional character or list. If you specify a string, then the string is used as the in-memory table name. A list can be used to specify properties for the in-memory table as follows: <ul style="list-style-type: none"> name An optional character that specifies the name for the in-memory table. By default, the name of the data frame is used. caslib An optional character that specifies the caslib. Specify this parameter to override the active caslib. label An optional character that specifies a descriptive label for the data. replace An optional logical. When set to TRUE, you can replace an existing in-memory table with the same name in the same caslib. The default value is FALSE. promote An optional logical. When set to TRUE, the in-memory table has global scope and can be available to other CAS sessions (subject to access controls). The default value is FALSE and the in-memory table has session scope so that it is accessible with the session that uploaded the table only. Session-scope tables are ideal for data analysis. Global-scope tables are better suited for reporting. replication An optional numeric that specifies the number of redundant copies of in-memory blocks. This parameter applies to distributed servers only. The default value is 1.

Value

[CASTable](#)

See Also

Other functions for loading in-memory data: [cas.read.csv\(\)](#), [cas.read.jump\(\)](#), [cas.read.sas7bdat\(\)](#), [cas.read.xlsx\(\)](#), [cas.readRDS\(\)](#)

Examples

```
## Not run:
```

```
myCasTable <- cas.read.table(s, "/path/to/data.tsv", header=TRUE,
                             sep="\t", casOut=list(name="mycastable"))

## End(Not run)
```

cas.read.xlsx

Read an XLSX File and Upload to a CAS Table

Description

This function is a convenience wrapper for the R `read.xlsx` and `as.casTable` functions. After reading the file that is accessible to the R client, it is uploaded to an in-memory table in CAS (the server).

Usage

```
cas.read.xlsx(
  conn,
  file,
  sheetIndex = 1,
  sheetName = NULL,
  rowIndex = NULL,
  startRow = NULL,
  endRow = NULL,
  colIndex = NULL,
  as.data.frame = TRUE,
  header = TRUE,
  colClasses = NA,
  keepFormulas = FALSE,
  encoding = "unknown",
  casOut = list(name = "", replace = FALSE)
)
```

Arguments

<code>conn</code>	An instance of a CAS object that represents a connection and CAS session.
<code>file</code>	An character string that specifies the filename for the XLSX file. This value is passed to <code>read.xlsx</code> .
<code>sheetIndex</code>	An optional numeric that specifies the sheet in the workbook. This value is passed to <code>read.xlsx</code> .
<code>sheetName</code>	An optional character string that specifies the sheet name in the workbook. This value is passed to <code>read.xlsx</code> .
<code>rowIndex</code>	An optional numeric vector that specifies the rows to read from the workbook. By default, all rows are read. This value is passed to <code>read.xlsx</code> .
<code>startRow</code>	An optional numeric that specifies the first row of the workbook to read. This value is ignored if <code>rowIndex</code> is specified. This value is passed to <code>read.xlsx</code> .

endRow	An optional numeric that specifies the last row of the workbook to read. This value is ignored if rowIndex is specified. This value is passed to read.xlsx.
colIndex	An optional numeric vector that specifies the variables to read from the workbook. By default, all variables are read. This value is passed to read.xlsx.
as.data.frame	An optional logical value that specifies whether the data should be coerced into a data frame. This value is passed to read.xlsx.
header	An optional logical that specifies whether the first line of the file contains variable names.
colClasses	An optional character vector that specifies the classes for the columns. This value is passed to read.xlsx.
keepFormulas	An optional logical value that specifies whether Excel formulas are included as text or if they are evaluated and the result is read as data. This value is passed to read.xlsx.
encoding	An optional character string that specifies the encoding for character data. This value is passed to read.xlsx.
casOut	An optional character or list. If you specify a string, then the string is used as the in-memory table name. A list can be used to specify properties for the in-memory table as follows: <ul style="list-style-type: none"> name An optional character that specifies the name for the in-memory table. By default, the name of the data frame is used. caslib An optional character that specifies the caslib. Specify this parameter to override the active caslib. label An optional character that specifies a descriptive label for the data. replace An optional logical. When set to TRUE, you can replace an existing in-memory table with the same name in the same caslib. The default value is FALSE. promote An optional logical. When set to TRUE, the in-memory table has global scope and can be available to other CAS sessions (subject to access controls). The default value is FALSE and the in-memory table has session scope so that it is accessible with the session that uploaded the table only. Session-scope tables are ideal for data analysis. Global-scope tables are better suited for reporting. replication An optional numeric that specifies the number of redundant copies of in-memory blocks. This parameter applies to distributed servers only. The default value is 1.

Value

[CASTable](#)

See Also

Other functions for loading in-memory data: [cas.read.csv\(\)](#), [cas.read.jmp\(\)](#), [cas.read.sas7bdat\(\)](#), [cas.read.table\(\)](#), [cas.readRDS\(\)](#)

Examples

```
## Not run:
myCasTable <- cas.read.xlsx(s, file="/path/to/data_out.xlsx",
  sheetIndex = 1,
  casOut=list(name="mycastable", replace=TRUE))

## End(Not run)
```

cas.readRDS

*Read an RDS File and Upload to a CAS Table***Description**

This function is a convenience wrapper for the R `readRDS` and `as.casTable` functions. After reading the file that is accessible to the R client, it is uploaded to an in-memory table in CAS (the server).

Usage

```
cas.readRDS(
  conn,
  file,
  refhook = NULL,
  casOut = list(name = "", replace = FALSE)
)
```

Arguments

<code>conn</code>	An instance of a CAS object that represents a connection and CAS session.
<code>file</code>	An character string that specifies the filename for the RDS file. This value is passed to <code>readRDS</code> .
<code>refhook</code>	An optional value that is passed to <code>readRDS</code> .
<code>casOut</code>	An optional character or list. If you specify a string, then the string is used as the in-memory table name. A list can be used to specify properties for the in-memory table as follows: <ul style="list-style-type: none"> <code>name</code> An optional character that specifies the name for the in-memory table. By default, the name of the data frame is used. <code>caslib</code> An optional character that specifies the caslib. Specify this parameter to override the active caslib. <code>label</code> An optional character that specifies a descriptive label for the data. <code>replace</code> An optional logical. When set to <code>TRUE</code>, you can replace an existing in-memory table with the same name in the same caslib. The default value is <code>FALSE</code>.

promote An optional logical. When set to TRUE, the in-memory table has global scope and can be available to other CAS sessions (subject to access controls). The default value is FALSE and the in-memory table has session scope so that it is accessible with the session that uploaded the table only. Session-scope tables are ideal for data analysis. Global-scope tables are better suited for reporting.

replication An optional numeric that specifies the number of redundant copies of in-memory blocks. This parameter applies to distributed servers only. The default value is 1.

Value

CASTable

See Also

Other functions for loading in-memory data: [cas.read.csv\(\)](#), [cas.read.jsp\(\)](#), [cas.read.sas7bdat\(\)](#), [cas.read.table\(\)](#), [cas.read.xlsx\(\)](#)

Examples

```
## Not run:
myCasTable <- cas.readRDS(s, file="/path/to/data_out.rds",
  casOut=list(name="mycastable"))

## End(Not run)
```

cas.saveRDS

Write a CAS Table to RDS

Description

This function downloads an in-memory table from the CAS server and saves it as an RDS file that is accessible to R (the client). This function is a convenience wrapper for the R `saveRDS` function.

Usage

```
cas.saveRDS(
  CASTable,
  file = "",
  ascii = FALSE,
  version = NULL,
  compress = TRUE,
  refhook = NULL
)
```

Arguments

CASTable	The instance of the CASTable to save as an RDS file.
file	An character string that specifies the filename for the RDS file. If you do not specify the file, then the in-memory table name is used with an .rds suffix. This value is passed to saveRDS.
ascii	An optional logical value. When set to TRUE or NA, then an ASCII representation is written. Otherwise, a binary is written. This value is passed to saveRDS.
version	An optional numeric value. This value is passed to saveRDS.
compress	An optional logical value. When set to one of TRUE, "gzip", "bzip2", or "xz", then compression is used. TRUE performs gzip. This value is passed to saveRDS.
refhook	An optional value that is passed to saveRDS.

See Also

Other functions for saving in-memory data: [cas.write.csv\(\)](#), [cas.write.table\(\)](#), [cas.write.xlsx\(\)](#)

Examples

```
## Not run:
cas.saveRDS(myCasTable, file="/path/to/data_out.rds")

## End(Not run)
```

cas.sd

Standard Deviation

Description

Returns the standard deviation of the values for each column in the input table.

Usage

```
cas.sd(CASTable, na.rm = TRUE)
```

Arguments

na.rm	An optional logical. When set to FALSE, missing values (NA) are not removed from the analysis. By default, missing values are ignored.
x	CASTable.

Details

This function operates on numeric columns only.

Value

casDataFrame

The result includes one row for each numeric variable and a column that is named Std for the standard deviation.

Examples

```
## Not run:  
cas.sd(ct[1:4])  
cas.sd(ct$n2)  
  
## End(Not run)
```

cas.st derr	<i>Standard Error</i>
-------------	-----------------------

Description

Returns the standard error of the values for each column in the input table.

Usage

```
cas.st derr(CASTable)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

vector

The result is a named numeric vector. You can access the standard error by column name or index.

Examples

```
## Not run:  
x <- cas.st derr(irisct)  
x['Sepal.Length']  
x[1:2]  
  
## End(Not run)
```

cas.sum	<i>Column Sums</i>
---------	--------------------

Description

Returns the sum of the values for each column in the input table.

Usage

```
cas.sum(CASTable)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

casDataFrame

The result includes one row for each numeric variable and a column that is named Sum for the summed value.

See Also

colSums, CASTable-method

Examples

```
## Not run:  
cas.sum(ct[1:4])  
cas.sum(ct$n2)  
  
## End(Not run)
```

cas.terminate	<i>End a CAS session and close the connection</i>
---------------	---

Description

End a CAS session and close the connection

Usage

```
cas.terminate(conn)
```

Arguments

CAS	The CAS connection object
-----	---------------------------

cas.tvalue	<i>T-Statistics for Hypothesis Testing</i>
------------	--

Description

Returns the t-statistic for the values for each column in the input table.

Usage

```
cas.tvalue(CASTable)
```

Arguments

x	CASTable.
---	-----------

Details

This function operates on numeric columns only.

Value

vector

The result is a named numeric vector. You can access the t-statistic by column name or index.

Examples

```
## Not run:  
x <- cas.tvalue(irisct)  
x['Sepal.Length']  
x[1:2]  
  
## End(Not run)
```

cas.upload	<i>Upload a data.frame or file to a CAS table</i>
------------	---

Description

Upload a data.frame or file to a CAS table

Usage

```
cas.upload(conn, ...)
```

Arguments

...	Optional parameters that are passed to the table.loadtable action.
CAS	The CAS connection object
data.frame/character	The data.frame, filename, or URL to upload.

Value

list

cas.upload.file	<i>Upload a data file to a CAS table</i>
-----------------	--

Description

Upload a data file to a CAS table

Usage

```
cas.upload.file(conn, ...)
```

Arguments

...	Optional parameters that are passed to the table.loadtable action.
CAS	The CAS connection object
character	The filename to upload

Value

[CASTable](#)

cas.upload.frame	<i>Upload a data.frame to a CAS table</i>
------------------	---

Description

Upload a data.frame to a CAS table

Usage

```
cas.upload.frame(conn, ...)
```

Arguments

...	Optional parameters that are passed to the table.loadtable action.
CAS	The CAS connection object
data.frame	The data.frame to upload

Value

[CASTable](#)

cas.uss	<i>Uncorrected Sum of Squares</i>
---------	-----------------------------------

Description

Returns the uncorrected sum of squares of the values for each column in the input table.

Usage

```
cas.uss(CASTable)
```

Arguments

x	CASTable.
---	-----------

Details

This function operates on numeric columns only.

Value

vector

The result is a named numeric vector. You can access the uncorrected sum of squares by column name or index.

Examples

```
## Not run:  
x <- cas.uss(irisct)  
x['Sepal.Length']  
x[1:2]  
  
## End(Not run)
```

cas.var	<i>Variance</i>
---------	-----------------

Description

Returns the variance of the values for each column in the input table.

Usage

```
cas.var(CASTable)
```

Arguments

x CASTable.

Details

This function operates on numeric columns only.

Value

casDataFrame

The result includes one row for each numeric variable and a column that is named Var for the variance.

Examples

```
## Not run:  
cas.var(ct[1:4])  
cas.var(ct$n2)  
  
## End(Not run)
```

cas.write.csv	<i>Write a CAS Table to a CSV File</i>
---------------	--

Description

This function downloads an in-memory table from the CAS server and saves it as a CSV file that is accessible to R (the client). This function is a convenience wrapper for the R `write.csv` function.

Usage

```
cas.write.csv(  
  CASTable,  
  file = "",  
  quote = TRUE,  
  eol = "\n",  
  na = "NA",  
  row.names = TRUE,  
  fileEncoding = ""  
)
```

Arguments

CASTable	The instance of the CASTable to save as as a CSV file.
file	An character string that specifies the filename for the CSV file. If you do not specify the file, then the in-memory table name is used with a .csv suffix. This value is passed to <code>write.csv</code> .
quote	An optional logical value or numeric vector. This value is passed to <code>write.csv</code> .
eol	An optional character string that is used as the end-of-line character or characters. This value is passed to <code>write.csv</code> .
na	An optional character string to represent missing values. This value is passed to <code>write.csv</code> .
row.names	An optional logical value or a character vector of row names. This value is passed to <code>write.csv</code> .
fileEncoding	An optional character string that specifies the encoding to use for writing the CSV file. This value is passed to <code>write.csv</code> .
append	An optional logical value This value is passed to <code>write.csv</code> .
sep	An optional character that is used to separate values in the CSV file. This value is passed to <code>write.csv</code> .
dec	An optional character to represent the decimal separator. This value is passed to <code>write.csv</code> .
col.names	An optional logical value or a character vector of column names. This value is passed to <code>write.csv</code> .
qmethod	An optional chracter string that describes how to write embedded quotation marks. This value is passed to <code>write.csv</code> .

Details

This function saves the file on the R client. As an alternative, you can use the `cas.table.save` generated function to save a server-side CSV file.

See Also

Other functions for saving in-memory data: [cas.saveRDS\(\)](#), [cas.write.table\(\)](#), [cas.write.xlsx\(\)](#)

Examples

```
## Not run:
# upload a SAS data set to an in-memory table
gold_medals <- cas.read.sas7bdat(s, "/path/to/gold_medals.sas7bdat")

# download the in-memory table as a CSV file
cas.write.csv(gold_medals, "~/gold_medals.csv")

## End(Not run)
```

`cas.write.csv2`*Write a CAS Table to a CSV File*

Description

This function is identical to [cas.write.csv](#) except that it wraps the R `write.csv2` function. The `write.csv2` function uses a comma for the decimal separator and a semicolon for the field delimiter.

Usage

```
cas.write.csv2(
  CASTable,
  file = "",
  quote = TRUE,
  eol = "\n",
  na = "NA",
  row.names = TRUE,
  fileEncoding = ""
)
```

<code>cas.write.table</code>	<i>Write a CAS Table to a Table</i>
------------------------------	-------------------------------------

Description

This function downloads an in-memory table from the CAS server and saves it as a file that is accessible to R (the client). This function is a convenience wrapper for the `Rwrite.table` function.

Usage

```
cas.write.table(
  CASTable,
  file = "",
  append = FALSE,
  quote = TRUE,
  sep = " ",
  eol = "\n",
  na = "NA",
  dec = ".",
  row.names = TRUE,
  col.names = TRUE,
  qmethod = c("escape", "double"),
  fileEncoding = ""
)
```

Arguments

<code>CASTable</code>	The instance of the <code>CASTable</code> to save as a file.
<code>file</code>	An character string that specifies the filename for the file. If you do not specify the file, then the table is printed to the terminal.
<code>append</code>	An optional logical value. This value is passed to <code>write.table</code> .
<code>quote</code>	An optional logical value or numeric vector. This value is passed to <code>write.table</code> .
<code>sep</code>	An optional character that is used to separate values in the file. This value is passed to <code>write.table</code> .
<code>eol</code>	An optional character string that is used as the end-of-line character or characters. This value is passed to <code>write.table</code> .
<code>na</code>	An optional character string to represent missing values. This value is passed to <code>write.table</code> .
<code>dec</code>	An optional character to represent the decimal separator. This value is passed to <code>write.table</code> .
<code>row.names</code>	An optional logical value or a character vector of row names. This value is passed to <code>write.table</code> .
<code>col.names</code>	An optional logical value or a character vector of column names. This value is passed to <code>write.table</code> .

qmethod	An optional character string that describes how to write embedded quotation marks. This value is passed to <code>write.table</code> .
fileEncoding	An optional character string that specifies the encoding to use for writing the file. This value is passed to <code>write.table</code> .

See Also

Other functions for saving in-memory data: [cas.saveRDS\(\)](#), [cas.write.csv\(\)](#), [cas.write.xlsx\(\)](#)

Examples

```
## Not run:
#
cas.write.table(myCasTable, file="/path/to/data_out.txt", na="")
## End(Not run)
```

cas.write.xlsx	<i>Write a CAS Table to a Microsoft Excel Workbook</i>
----------------	--

Description

This function downloads an in-memory table from the CAS server and saves it as an XLSX file that is accessible to R (the client). This function is a convenience wrapper for the R `write.xlsx` function.

Usage

```
cas.write.xlsx(
  CASTable,
  file = "",
  sheetName = "Sheet1",
  col.names = TRUE,
  row.names = TRUE,
  append = FALSE,
  showNA = TRUE
)
```

Arguments

CASTable	The instance of the CASTable to save as a CSV file.
file	An character string that specifies the filename for the XLSX file. If you do not specify the file, then the in-memory table name is used with an <code>.xlsx</code> suffix. This value is passed to <code>write.xlsx</code> .
sheetName	An optional character string that specifies the sheet name in the workbook. This value is passed to <code>write.xlsx</code> .

col.names	An optional logical value that specifies whether column names are included in the workbook. This value is passed to write.xlsx.
row.names	An optional logical value that specifies whether row names are included in the workbook. This value is passed to write.xlsx.
append	An optional logical value. This value is passed to write.xlsx.
showNA	An optional logical value. This value is passed to write.xlsx.

See Also

Other functions for saving in-memory data: [cas.saveRDS\(\)](#), [cas.write.csv\(\)](#), [cas.write.table\(\)](#)

Examples

```
## Not run:  
cas.write.xlsx(myCasTable, file="/path/to/data_out.xlsx")  
  
## End(Not run)
```

casDataFrame-class *CAS Data Frame Class*

Description

A casDataFrame is a superset of data.frame. The class is used primarily by functions within the package to store tabular data from CAS action results and to associate CAS metadata with the tabular data.

Usage

```
casDataFrame(  
  ...,  
  name = "",  
  label = "",  
  title = "",  
  attrs = list(),  
  col.labels = "",  
  col.formats = "",  
  col.attrs = list(),  
  col.sizes = list(),  
  col.types = "",  
  col.widths = 0  
)
```

Details

A casDataFrame instance is a data object in R (the client).

Value

casDataFrame

Slots

name An optional character string that specifies CAS metadata for a table name.

label An optional character string that specifies CAS metadata for a table label.

title An optional character string that specifies CAS metadata for a table title.

attrs An optional list of key-value pairs the specify user-defined CAS metadata.

col.labels An optional character string that specifies CAS metadata for column labels.

col.formats An optional character string that specifies CAS metadata for column formats.

col.attrs An optional list that specifies CAS metadata for column attributes.

col.sizes An optional list that specifies CAS metadata for the number of bytes in the widest row.

col.types An optional character that specifies CAS metadata for the column data types.

col.widths An optional numeric that specifies CAS metadata for column widths.

df The data.frame to encapsulate in the casDataFrame.

See Also

casDataFrame

CASTable-class

CAS Table Object

Description

CAS Table Object

Value

CASTable

Slots

conn A [CAS](#) object that represents the connection and session on the server.

tname An optional character string for the table name.

caslib An optional character string that identifies the caslib for the in-memory table. Specify this parameter to override the active caslib.

where An optional character string that specifies a filter for the rows to process. The filter uses syntax that is specific to SAS.

- orderby An optional list of column names. Rows are partitioned according to the columns in the groupby parameter and then ordered according to the values of the columns specified in this parameter.
- groupby An optional list of column names. If you specify this parameter when you load an in-memory table, then the table is partitioned by the columns. If you specify this parameter when running an action, then BY-groups are formed temporarily for the duration of the action.
- gbmode An optional character string. Values are NOSORT (default) or REDISTRIBUTE. See the CAS product documentation for more information.
- computedOnDemand An optional logical flag that indicates whether the computed variables are created when the table is loaded (False) or to compute them when an action begins (True).
- computedVars An optional character string list that identifies the name and optional information such as a format and label.
- computedVarsProgram An optional character string list. Specify the expression to use for computing each of the computed variables.
- names An optional list of column names.

See Also

as.casTable, defCasTable

cbind.casTable	<i>Combine CAS Tables by Columns</i>
----------------	--------------------------------------

Description

Combine CAS Tables by Columns

Usage

```
## S3 method for class 'casTable'
cbind(..., deparse.level = 1)
```

cbind2.casTable	<i>Combine CAS Tables by Columns</i>
-----------------	--------------------------------------

Description

This is the implementation of cbind for in-memory tables.

Usage

```
cbind2.casTable(x, y, ...)

## S4 method for signature 'CASTable,ANY'
cbind2(x, y, ...)
```

Arguments

... Arguments that are passed to method arguments.
deparse.level See the help for base::cbind.

Value

CASTable

Examples

```
## Not run:  
cbind(ct1, ct2)  
cbind(ct1[1:3], ct2$X1)  
  
## End(Not run)
```

colMeans,CASTable-method

Column Means

Description

Column Means

Usage

```
## S4 method for signature 'CASTable'  
colMeans(x)
```

Arguments

x CASTable.

Value

vector

See Also

cas.mean

Examples

```
## Not run:  
colMeans(ct[1:4])  
colMeans(ct$X1)  
  
## End(Not run)
```

colnames,CASTable-method

Column Names in a CAS Table

Description

Returns the column names from the in-memory table that is referenced by the [CASTable](#) object.

Usage

```
## S4 method for signature 'CASTable'  
colnames(x)
```

Arguments

x A [CASTable](#) object.

Value

vector

Note

You cannot use this function to set the column names.

See Also

names,CASTable-method

Examples

```
## Not run:  
colnames(ct1)  
colnames(ct[1:4])  
  
## End(Not run)
```

colSums,CASTable-method

Column Sums

Description

Column Sums

Usage

```
## S4 method for signature 'CASTable'  
colSums(x, na.rm = FALSE, dims = 1, ...)
```

Arguments

x CASTable.

Value

vector

See Also

cas.sum

Examples

```
## Not run:  
colSums(ct[1:4])  
colSums(ct$X1)  
  
## End(Not run)
```

cor,CASTable-method

Correlation

Description

Unlike the cor function for data frames, this function does not support specifying the method as "kendall" or "spearman." The results for method "pearson" are returned.

Usage

```
## S4 method for signature 'CASTable'  
cor(x, y = NULL, use = "everything", method = c("pearson"))
```

Arguments

x CASTable.

Value

matrix

Examples

```
## Not run:
cor(ct1)
cor(i2[1:4])
cor(i2[c('col1', 'col2', 'col3')])
cor(i2$col1, i2$col2)

## End(Not run)
```

cov,CASTable-method *Covariance*

Description

Unlike the cov function for data frames, this function does not support specifying the method as "kendall" or "spearman." The results for method "pearson" are returned.

Usage

```
## S4 method for signature 'CASTable'
cov(x, y = NULL, use = "everything", method = c("pearson"))
```

Arguments

x CASTable.

Value

matrix

Examples

```
## Not run:
cov(ct1)
cov(i2[1:4])
cov(i2[c('col1', 'col2', 'col3')])
cov(i2$col1, i2$col2)

## End(Not run)
```

defCasTable

*Create a CASTable Object for an Existing CAS Table***Description**

Creates a [CASTable](#) object to reference an existing in-memory table in CAS. You can use this function to reference tables that were loaded by other SAS products, other scripts, or from server-side loads with the `cas.table.loadTable` function.

Usage

```
defCasTable(
  conn,
  tablename,
  caslib = "",
  columns = "",
  where = "",
  orderby = list(),
  groupby = list(),
  gbmode = ""
)
```

Arguments

conn	A CAS object that represents a connection and session in CAS.
tablename	A character that specifies the in-memory table name. You can run the <code>cas.table.tableInfo</code> function to list the in-memory tables.
caslib	An optional character string that identifies the caslib for the in-memory table. Specify this parameter to override the active caslib.
columns	A list of column names.
where	A character string that specifies a filter for the rows to process. The filter uses syntax that is specific to SAS.
orderby	A list of column names. Rows are partitioned according to the columns in the groupby parameter and then ordered according to the values of the columns specified in this parameter.
groupby	A list of column names. If you specify this parameter when you load an in-memory table, then the table is partitioned by the columns. If you specify this parameter when running an action, then BY-groups are formed temporarily for the duration of the action.
gbmode	A character string. Values are NOSORT (default) or REDISTRIBUTE. See the CAS product documentation for more information.

Value[CASTable](#)

Examples

```
## Not run:
irisct <- as.casTable(s, iris, casOut="irisct")

# Create another CASTable instance to the same in-memory table,
# but specify that CAS actions are performed by groups of species.
irisct.grouped <- defCasTable(s, tablename="irisct", groupby=list("species"))

## End(Not run)
```

dim,CASTable-method *Dimensions of a CAS Table*

Description

Returns the number of rows and columns for the in-memory table that is referenced by the [CASTable](#) object.

Usage

```
## S4 method for signature 'CASTable'
dim(x)
```

Arguments

x A CASTable object.

Value

vector

Examples

```
## Not run:
dim(ct1)

## End(Not run)
```

dimnames, CASTable-method

Dimension Names of a CAS Table

Description

Dimension Names of a CAS Table

Usage

```
## S4 method for signature 'CASTable'  
dimnames(x)
```

Arguments

x A [CASTable](#) object.

Value

list

Examples

```
## Not run:  
dimnames(ct1)  
dimnames(ct[2:4])  
  
## End(Not run)
```

download_sas_binaries *Download SAS SWAT binary libraries for the current platform*

Description

Downloads and extracts platform-specific SWAT (SAS Scripting Wrapper for Analytics Transfer) binary libraries into a `swat/` subdirectory of the first library path returned by `‘.libPaths()’`.

Usage

```
download_sas_binaries(libpath = .libPaths())
```

Arguments

libpath Character vector of library paths. The first element is used as the installation root. Defaults to `‘.libPaths()’`.

Value

Invisibly returns the destination path of the extracted ‘libs‘ directory.

Examples

```
## Not run:  
download_sas_binaries()  
  
## End(Not run)
```

dropTable	<i>Remove a CAS Table</i>
-----------	---------------------------

Description

Drops the in-memory table on the server that is referenced by the [CASTable](#) object.

Usage

```
dropTable(x)
```

Arguments

x A CASTable object.

Note

This function drops the in-memory table but does not affect the original source file that the in-memory table was loaded from.

Examples

```
## Not run:  
dropTable(ct1)  
  
## End(Not run)
```

head, CASTable-method *Return the First Part of a CAS Table*

Description

Returns the first part of an in-memory table that is referenced by a [CASTable](#) object.

Usage

```
## S4 method for signature 'CASTable'
head(x, n = 6L)
```

Arguments

x A CASTable object.
n An optional positive integer that specifies the number of rows to return.

Value

A [casDataFrame](#) object with the first n rows.

Note

The head function is not deterministic between reloads of data or if nodes are added or removed from a distributed server.

Examples

```
## Not run:
head(ct1)
head(ct[1:4], 10)

## End(Not run)
```

is.castable *Test if an Object is a CAS Table*

Description

Test if an Object is a CAS Table

Usage

```
is.castable(x)
```

Arguments

x An R object.

Value

boolean

Examples

```
## Not run:  
is.castable(ct1)     # TRUE  
is.castable(iris)   # FALSE  
  
## End(Not run)
```

length,CASTable-method

Number of Columns in a CAS Table

Description

Returns the number of columns in an in-memory table that is referenced by the [CASTable](#) object.

Usage

```
## S4 method for signature 'CASTable'  
length(x)
```

Arguments

x A CASTable object.

Value

scalar

See Also

ncol,CASTable-method

Examples

```
## Not run:  
length(ct1)  
length(ct[1:4])  
  
## End(Not run)
```

listActionParms	<i>List CAS Action Parameters by Name</i>
-----------------	---

Description

This function displays the parameters for the specified CAS action.

Usage

```
listActionParms(conn, actn, display = TRUE)
```

Arguments

conn	An instance of a CAS object that represents a connection and CAS session.
actn	A string value that specifies the action name. You can specify the following forms: <ul style="list-style-type: none"> • <i>action-set.action-name</i> • <i>action-name</i> • <i>cas.action-set.action-name</i> <p>The third form matches the generated functions for the CAS actions.</p>
display	Should the parameters be printed to the screen?

Examples

```
## Not run:
# specify the action set and action name
listActionParms(conn, actn="simple.summary")

# fetch is in the table action set
listActionParms(s, actn="fetch")

# specify the generated function name
listActionParms(s, cas.regression.logistic)

## End(Not run)
```

listActionSets	<i>List CAS Action Sets</i>
----------------	-----------------------------

Description

This function lists all action sets that are installed on the CAS server. The results include a column that is named loaded to indicate if the action set is already available for use or must be loaded before it can be used.

Usage

```
listActionSets(conn)
```

Arguments

conn An instance of a CAS object that represents a connection and CAS session.

Examples

```
## Not run:  
listActionSets(conn)  
  
## End(Not run)
```

loadActionSet	<i>Load a CAS Action Set</i>
---------------	------------------------------

Description

This function loads a CAS action set and generate an R function for each action.

Usage

```
loadActionSet(conn, actionSet = "")
```

Arguments

conn An instance of a CAS object that represents a connection and CAS session.
actionSet A string value that specifies the action set to load.

Examples

```
## Not run:  
loadActionSet(conn, actionSet="decisionTree")  
loadActionSet(conn, actionSet="regression")  
  
## End(Not run)
```

max,CASTable-method *Maximum Values*

Description

Maximum Values

Usage

```
## S4 method for signature 'CASTable'  
max(x)
```

Arguments

x CASTable.

Value

scalar

See Also

cas.max

Examples

```
## Not run:  
max(ct1$X1)  
max(ct1[[1:2]])  
  
## End(Not run)
```

mean,CASTable-method *Mean Value for a Single Column*

Description

Returns the mean value for the specified column in the input table.

Usage

```
## S4 method for signature 'CASTable'  
mean(x, ...)
```

Arguments

x CASTable.

Value

scalar

See Also

cas.mean

Examples

```
## Not run:  
mean(ct1$x1)  
mean(ct1[2])  
  
## End(Not run)
```

median, CASTable-method

Median Values

Description

Median Values

Usage

```
## S4 method for signature 'CASTable'  
median(x)
```

Arguments

x CASTable.

Value

scalar

See Also

cas.median

Examples

```
## Not run:  
median(ct1$x1)  
median(ct1[1:4])  
  
## End(Not run)
```

min,CASTable-method *Minimum Value*

Description

Minimum Value

Usage

```
## S4 method for signature 'CASTable'  
min(x)
```

Arguments

x CASTable.

Value

scalar

See Also

cas.min

Examples

```
## Not run:  
min(ct1$X1)  
min(ct1[[1:2]])  
  
## End(Not run)
```

names,CASTable-method *Names of a CAS Table*

Description

Returns the list of column names for the in-memory table that is referenced by the [CASTable](#) object.

Usage

```
## S4 method for signature 'CASTable'  
names(x)
```

Arguments

x A CASTable object.

Value

vector

Examples

```
## Not run:  
names(ct1)  
  
## End(Not run)
```

ncol,CASTable-method *Number of Columns in a CAS Table*

Description

Returns the number of columns in an in-memory table that is referenced by the [CASTable](#) object.

Usage

```
## S4 method for signature 'CASTable'  
ncol(x)
```

Arguments

x A CASTable object.

Value

scalar

See Also

length,CASTable-method

Examples

```
## Not run:  
ncol(ct1)  
ncol(ct[1:4])  
  
## End(Not run)
```

`nrow,CASTable-method` *Number of Rows in a CAS Table*

Description

Returns the number of rows in an in-memory table that is referenced by the [CASTable](#) object.

Usage

```
## S4 method for signature 'CASTable'
nrow(x)
```

Arguments

`x` A [CASTable](#) object.

Value

scalar

Examples

```
## Not run:
nrow(ct1)
nrow(ct[1:4])

## End(Not run)
```

`rbind,CASTable-method` *Combine CAS Tables by Rows*

Description

This is the implementation of `rbind` for in-memory tables.

Usage

```
## S4 method for signature 'CASTable'
rbind(..., deparse.level = 1)
```

Arguments

`...` Arguments that are passed to method arguments.
`deparse.level` See the help for `base::rbind`.

Value

CASTable

Examples

```
## Not run:
rbind(ct1,ct2)
rbind(ct1,ct2, ct3)

## End(Not run)
```

rbind.casTable	<i>Combine CAS Tables by Columns</i>
----------------	--------------------------------------

Description

Combine CAS Tables by Columns

Usage

```
rbind.casTable(..., deparse.level = 1)
```

rbind2.casTable	<i>Combine CAS Tables by Rows</i>
-----------------	-----------------------------------

Description

This is the implementation of rbind for in-memory tables.

Usage

```
rbind2.casTable(x, y, ...)
```

```
## S4 method for signature 'CASTable,ANY'
rbind2(x, y, ...)
```

Arguments

... Arguments that are passed to method arguments.

deparse.level See the help for base::rbind.

Value

CASTable

Examples

```
## Not run:  
rbind(ct1,ct2)  
rbind(ct1,ct2, ct3)  
  
## End(Not run)
```

rownames,CASTable-method

Row Names of a CAS Table

Description

Row Names of a CAS Table

Usage

```
## S4 method for signature 'CASTable'  
rownames(x)
```

Arguments

x A [CASTable](#) object.

Value

list of strings

Examples

```
## Not run:  
rownames(ct1)  
rownames(ct[2:4])  
  
## End(Not run)
```

runAction

Run a CAS Action by Name

Description

This function enables you to specify a CAS action name and run it. This is an alternative to running an action from the generated function names. An example of a generated function name is `cas.table.tableInfo`.

Usage

```
runAction(CASorCASTab = "", actn, check_errors = FALSE, ...)
```

Arguments

CASorCASTab	An instance of a CAS object that represents a connection and CAS session, or an instance of a CASTable.
actn	A character string that specifies the action set and action name to run.
...	Parameters that are passed to the CAS action.

Examples

```
## Not run:
# display the active caslib for the session
runAction(conn, "sessionProp.getSessOpt", name="caslib")

# display simple summary statistics for an uploaded data frame
mtcarsct <- as.casTable(conn, mtcars)
runAction(mtcarsct, "simple.summary")

# the preceding runAction function is functionally
# equivalent to the following generated function
cas.simple.summary(mtcarsct)

## End(Not run)
```

show,casDataFrame-method

CAS data frame show method

Description

CAS data frame show method

Usage

```
## S4 method for signature 'casDataFrame'
show(object)
```

Arguments

object	A casDataFrame object.
--------	------------------------

Value

data frame listing

subset.casTable *Return a Subset of Rows and Columns from a CAS Table*

Description

Return a subset of rows and columns from a [CASTable](#) that meet subsetting criteria.

Usage

```
subset.casTable(x, subset, select = NULL, drop = FALSE, ...)
```

Arguments

x A CASTable object.

Value

A CASTable object with the rows and columns that meet the subset criteria.

Examples

```
## Not run:
subset(ct, subset = ct[4] > 15, select = c("n1", "n4", "s"), drop = FALSE)
subset(ct, subset = ct$n4 > 15, select = c(1, 4, 5), drop = FALSE)

## End(Not run)
```

summary,CASTable-method
Summary Statistics

Description

Returns simple descriptive statistics.

Usage

```
## S4 method for signature 'CASTable'
summary(object, maxsum = 7, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

object CASTable.

Value

table

Examples

```
## Not run:
summary(ct1)
summary(ct1[[1:4]])

## End(Not run)
```

swat

SWAT: SAS Wrapper for Analytics Transfer (SWAT)

Description

This package enables you to connect from R to a SAS Cloud Analytic Services host, run actions on in-memory tables, and work with the results of the actions.

- The [CAS](#) class provides an interface to your connection to the CAS server and CAS session.
- The [CASTable](#) class provides an interface to the in-memory tables.
- The [casDataFrame](#) class provides an interface to the results for most actions.

Details

Depending on how you install the package, you might be able to use binary communication with CAS. This is more efficient for bandwidth, but requires that your R installation have access to a precompiled library (rswat.so or rswat.dll). An alternative is to communicate with the server using the REST interface of the server over HTTP. See the connection examples that follow.

The responses and results of the actions are returned as R objects.

Connect and start a session

```
s <- CAS('cloud.example.com', 5570) # binary communication

s2 <- CAS('cloud.example.com', 8777, protocol='https')
```

Run a simple action

```
results <- runAction(s, "builtins.serverStatus")
results$server
nodes actions
1      1      15
```

You can also run an action using the generated R function:

```
results <- cas.builtins.serverStatus(s)
results$server
nodes actions
1      1      15
```

Upload a dataframe to a CASTable

```
irisct <- as.casTable(s, iris)
```

Load a CAS actionSet

```
runAction(s, "builtins.loadActionSet", actionSet="regression")
```

Useful links

- <http://developer.sas.com/guides/r.html>
- <https://github.com/sassoftware/r-swat>
- Enter issues at <https://github.com/sassoftware/r-swat/issues>

Action documentation

See the [Actions and Action Sets by Name and Product](#)

Author(s)

Maintainer: Jared Dean <Jared.Dean@sas.com>

Authors:

- Tom Weber <Tom.Weber@sas.com>
- Kevin Smith <Kevin.Smith@sas.com>

tail,CASTable-method *Return the Last Part of a CAS Table*

Description

Returns the last part of an in-memory table that is referenced by a [CASTable](#) object.

Usage

```
## S4 method for signature 'CASTable'  
tail(x, n = 6L)
```

Arguments

x A CASTable object.

Value

A [casDataFrame](#) object with the last n rows.

Note

The tail function is not deterministic between reloads of data or if nodes are added or removed from a distributed server.

Examples

```
## Not run:  
tail(ct1)  
tail(ct[1:4], 10)  
  
## End(Not run)
```

to.casDataFrame

Convert a CAS Table to a CAS Data Frame (Download)

Description

Downloads the in-memory table that is referenced by the CASTable object and stores it as a casDataFrame in R. This function is used primarily by the package to store the results of a CAS action.

Usage

```
to.casDataFrame(ct, obs = 32768)
```

Arguments

ct The CASTable object to download.

Value

Returns a casDataFrame object that contains a copy of the in-memory data.

Examples

```
## Not run:  
cdf = to.casDataFrame(CASTable)  
  
## End(Not run)
```

to.data.frame	<i>Convert a CAS data frame to an R Data Frame</i>
---------------	--

Description

This function returns the R data.frame object that is encapsulated in a casDataFrame. The CAS metadata is not available in the returned data.frame.

Usage

```
to.data.frame(cdf)
```

Arguments

cdf	The casDataFrame to convert
-----	-----------------------------

Value

Data Frame

Examples

```
## Not run:  
df3 = to.data.frame(cdf)  
  
## End(Not run)
```

to.r.data.frame	<i>Convert a CAS Table to a R Data Frame (Download)</i>
-----------------	---

Description

Downloads the in-memory table that is referenced by the CASTable object and stores it as a data.frame in R. This function is used to download datasets from CAS.

Usage

```
to.r.data.frame(ct, obs = 32768)
```

Arguments

ct	The CASTable object to download.
obs	Number of rows to download, by default 32768

Value

Returns a data.frame object that contains a copy of the in-memory data.

Examples

```
## Not run:  
rdf = to.r.data.frame(CASTable)  
  
## End(Not run)
```

unique, CASTable-method

Extract Unique Values from a CAS Table

Description

Extracts distinct values from columns in a [CASTable](#).

Usage

```
## S4 method for signature 'CASTable'  
unique(x, incomparables = FALSE, ...)
```

Arguments

x	A CASTable object.
incomparables	A vector of values that cannot be compared. See the help for base::unique.
...	Arguments that are passed to method arguments.

Value

A [casDataFrame](#) object.

Examples

```
## Not run:  
unique(ct[4:5])  
unique(ct$s)  
unique(ct[4])  
  
## End(Not run)
```

Index

!, CASTable-method ([, CASTable-method), 3
!=, ANY, CASTable-method
 ([, CASTable-method), 3
!=, CASTable, ANY-method
 ([, CASTable-method), 3
!=, CASTable, CASTable-method
 ([, CASTable-method), 3
* **functions for loading in-memory data**
 cas.read.csv, 21
 cas.read.jmp, 22
 cas.read.sas7bdat, 24
 cas.read.table, 25
 cas.read.xlsx, 28
 cas.readRDS, 30
* **functions for saving in-memory data**
 cas.saveRDS, 31
 cas.write.csv, 39
 cas.write.table, 41
 cas.write.xlsx, 42
*, ANY, CASTable-method
 ([, CASTable-method), 3
*, CASTable, ANY-method
 ([, CASTable-method), 3
*, CASTable, CASTable-method
 ([, CASTable-method), 3
+, ANY, CASTable-method
 ([, CASTable-method), 3
+, CASTable, ANY-method
 ([, CASTable-method), 3
+, CASTable, CASTable-method
 ([, CASTable-method), 3
-, ANY, CASTable-method
 ([, CASTable-method), 3
-, CASTable, ANY-method
 ([, CASTable-method), 3
-, CASTable, CASTable-method
 ([, CASTable-method), 3
.cas.arith ([, CASTable-method), 3
.cas.compare ([, CASTable-method), 3
.cas.logic ([, CASTable-method), 3
/, ANY, CASTable-method
 ([, CASTable-method), 3
/, CASTable, ANY-method
 ([, CASTable-method), 3
/, CASTable, CASTable-method
 ([, CASTable-method), 3
<, ANY, CASTable-method
 ([, CASTable-method), 3
<, CASTable, ANY-method
 ([, CASTable-method), 3
<, CASTable, CASTable-method
 ([, CASTable-method), 3
<=, ANY, CASTable-method
 ([, CASTable-method), 3
<=, CASTable, ANY-method
 ([, CASTable-method), 3
<=, CASTable, CASTable-method
 ([, CASTable-method), 3
==, ANY, CASTable-method
 ([, CASTable-method), 3
==, CASTable, ANY-method
 ([, CASTable-method), 3
==, CASTable, CASTable-method
 ([, CASTable-method), 3
>, ANY, CASTable-method
 ([, CASTable-method), 3
>, CASTable, ANY-method
 ([, CASTable-method), 3
>, CASTable, CASTable-method
 ([, CASTable-method), 3
>=, ANY, CASTable-method
 ([, CASTable-method), 3
>=, CASTable, ANY-method
 ([, CASTable-method), 3
>=, CASTable, CASTable-method
 ([, CASTable-method), 3
[, CASTable-method, 3
[<-, CASTable-method

- [, CASTable-method), 3
- [[, CASTable-method ([, CASTable-method), 3
- \$/, CASTable-method ([, CASTable-method), 3
- \$<- , CASTable-method ([, CASTable-method), 3
- %/%, ANY, CASTable-method ([, CASTable-method), 3
- %/%, CASTable, ANY-method ([, CASTable-method), 3
- %/%, CASTable, CASTable-method ([, CASTable-method), 3
- %%, ANY, CASTable-method ([, CASTable-method), 3
- %%, CASTable, ANY-method ([, CASTable-method), 3
- %%, CASTable, CASTable-method ([, CASTable-method), 3
- &, ANY, CASTable-method ([, CASTable-method), 3
- &, CASTable, ANY-method ([, CASTable-method), 3
- &, CASTable, CASTable-method ([, CASTable-method), 3
- ^, ANY, CASTable-method ([, CASTable-method), 3
- ^, CASTable, ANY-method ([, CASTable-method), 3
- ^, CASTable, CASTable-method ([, CASTable-method), 3
- as.casDataFrame, 7
- as.casTable, 8
- CAS, 8, 9, 44, 50, 67
- CAS (CAS-class), 10
- CAS Actions, 9
- CAS-class, 10
- cas.accessControl.assumeRole (CAS Actions), 9
- cas.accessControl.checkInAllObjects (CAS Actions), 9
- cas.accessControl.checkOutObject (CAS Actions), 9
- cas.accessControl.commitTransaction (CAS Actions), 9
- cas.accessControl.completeBackup (CAS Actions), 9
- cas.accessControl.createBackup (CAS Actions), 9
- cas.accessControl.deleteBWLlist (CAS Actions), 9
- cas.accessControl.dropRole (CAS Actions), 9
- cas.accessControl.isAuthorized (CAS Actions), 9
- cas.accessControl.isAuthorizedActions (CAS Actions), 9
- cas.accessControl.isAuthorizedColumns (CAS Actions), 9
- cas.accessControl.isAuthorizedTables (CAS Actions), 9
- cas.accessControl.isInRole (CAS Actions), 9
- cas.accessControl.listAcsActionSet (CAS Actions), 9
- cas.accessControl.listAcsData (CAS Actions), 9
- cas.accessControl.listAllPrincipals (CAS Actions), 9
- cas.accessControl.listMetadata (CAS Actions), 9
- cas.accessControl.operActionMd (CAS Actions), 9
- cas.accessControl.operActionSetMd (CAS Actions), 9
- cas.accessControl.operAdminMd (CAS Actions), 9
- cas.accessControl.operBWPPaths (CAS Actions), 9
- cas.accessControl.operColumnMd (CAS Actions), 9
- cas.accessControl.operTableMd (CAS Actions), 9
- cas.accessControl.removeAllAcsActionSet (CAS Actions), 9
- cas.accessControl.removeAllAcsData (CAS Actions), 9
- cas.accessControl.repAllAcsAction (CAS Actions), 9
- cas.accessControl.repAllAcsActionSet (CAS Actions), 9
- cas.accessControl.repAllAcsCaslib (CAS Actions), 9
- cas.accessControl.repAllAcsColumn (CAS Actions), 9

- cas.accessControl.repAllAcTable (CAS Actions), 9
- cas.accessControl.rollbackTransaction (CAS Actions), 9
- cas.accessControl.showRolesAllowed (CAS Actions), 9
- cas.accessControl.showRolesIn (CAS Actions), 9
- cas.accessControl.startTransaction (CAS Actions), 9
- cas.accessControl.statusTransaction (CAS Actions), 9
- cas.accessControl.updSomeAcAction (CAS Actions), 9
- cas.accessControl.updSomeAcActionSet (CAS Actions), 9
- cas.accessControl.updSomeAcCaslib (CAS Actions), 9
- cas.accessControl.updSomeAcColumn (CAS Actions), 9
- cas.accessControl.updSomeAcTable (CAS Actions), 9
- cas.accessControl.whatCheckoutsExist (CAS Actions), 9
- cas.accessControl.whatIsEffective (CAS Actions), 9
- cas.aggregation.aggregate (CAS Actions), 9
- cas.aStore.describe (CAS Actions), 9
- cas.aStore.download (CAS Actions), 9
- cas.aStore.score (CAS Actions), 9
- cas.aStore.upload (CAS Actions), 9
- cas.autotune.tuneDecisionTree (CAS Actions), 9
- cas.autotune.tuneFactMac (CAS Actions), 9
- cas.autotune.tuneForest (CAS Actions), 9
- cas.autotune.tuneGradientBoostTree (CAS Actions), 9
- cas.autotune.tuneNeuralNet (CAS Actions), 9
- cas.autotune.tuneSvm (CAS Actions), 9
- cas.bayesianNetClassifier.bnet (CAS Actions), 9
- cas.bglimmix.bglimmix (CAS Actions), 9
- cas.bioMedImage.buildSurface (CAS Actions), 9
- cas.boolRule.brScore (CAS Actions), 9
- cas.boolRule.brTrain (CAS Actions), 9
- cas.builtins.about (CAS Actions), 9
- cas.builtins.actionSetInfo (CAS Actions), 9
- cas.builtins.addNode (CAS Actions), 9
- cas.builtins.casCommon (CAS Actions), 9
- cas.builtins.echo (CAS Actions), 9
- cas.builtins.getLicensedProductInfo (CAS Actions), 9
- cas.builtins.getLicenseInfo (CAS Actions), 9
- cas.builtins.help (CAS Actions), 9
- cas.builtins.history (CAS Actions), 9
- cas.builtins.httpAddress (CAS Actions), 9
- cas.builtins.installActionSet (CAS Actions), 9
- cas.builtins.listNodes (CAS Actions), 9
- cas.builtins.loadActionSet (CAS Actions), 9
- cas.builtins.log (CAS Actions), 9
- cas.builtins.modifyQueue (CAS Actions), 9
- cas.builtins.ping (CAS Actions), 9
- cas.builtins.queryActionSet (CAS Actions), 9
- cas.builtins.queryName (CAS Actions), 9
- cas.builtins.reflect (CAS Actions), 9
- cas.builtins.refreshLicense (CAS Actions), 9
- cas.builtins.removeNode (CAS Actions), 9
- cas.builtins.serverStatus (CAS Actions), 9
- cas.builtins.shutdown (CAS Actions), 9
- cas.builtins.userInfo (CAS Actions), 9
- cas.cardinality.summarize (CAS Actions), 9
- cas.cdm.cdm (CAS Actions), 9
- cas.close, 11
- cas.clustering.kClus (CAS Actions), 9
- cas.conditionalRandomFields.crfScore (CAS Actions), 9
- cas.conditionalRandomFields.crfTrain (CAS Actions), 9
- cas.configuration.getServOpt (CAS Actions), 9
- cas.configuration.listServOpts (CAS Actions), 9

- cas.configuration.setServOpt (CAS Actions), 9
- cas.copula.copulaSimulate (CAS Actions), 9
- cas.count, 11
- cas.coureg.couregFitModel (CAS Actions), 9
- cas.coureg.couregViewStore (CAS Actions), 9
- cas.css, 12
- cas.cv, 13
- cas.dataDiscovery.profile (CAS Actions), 9
- cas.dataPreprocess.binning (CAS Actions), 9
- cas.dataPreprocess.catTrans (CAS Actions), 9
- cas.dataPreprocess.discretize (CAS Actions), 9
- cas.dataPreprocess.highCardinality (CAS Actions), 9
- cas.dataPreprocess.histogram (CAS Actions), 9
- cas.dataPreprocess.impute (CAS Actions), 9
- cas.dataPreprocess.kde (CAS Actions), 9
- cas.dataPreprocess.outlier (CAS Actions), 9
- cas.dataPreprocess.rustats (CAS Actions), 9
- cas.dataPreprocess.transform (CAS Actions), 9
- cas.dataStep.runCode (CAS Actions), 9
- cas.dataStep.runCodeTable (CAS Actions), 9
- cas.decisionTree.dtreeCode (CAS Actions), 9
- cas.decisionTree.dtreeMerge (CAS Actions), 9
- cas.decisionTree.dtreePrune (CAS Actions), 9
- cas.decisionTree.dtreeScore (CAS Actions), 9
- cas.decisionTree.dtreeSplit (CAS Actions), 9
- cas.decisionTree.dtreeTrain (CAS Actions), 9
- cas.decisionTree.forestCode (CAS Actions), 9
- cas.decisionTree.forestScore (CAS Actions), 9
- cas.decisionTree.forestTrain (CAS Actions), 9
- cas.decisionTree.gbtreeCode (CAS Actions), 9
- cas.decisionTree.gbtreeScore (CAS Actions), 9
- cas.decisionTree.gbtreeTrain (CAS Actions), 9
- cas.deepLearn.addLayer (CAS Actions), 9
- cas.deepLearn.buildModel (CAS Actions), 9
- cas.deepLearn.dlExportModel (CAS Actions), 9
- cas.deepLearn.dlImportModelWeights (CAS Actions), 9
- cas.deepLearn.dlLabelTarget (CAS Actions), 9
- cas.deepLearn.dlScore (CAS Actions), 9
- cas.deepLearn.dlTrain (CAS Actions), 9
- cas.deepLearn.dlTune (CAS Actions), 9
- cas.deepLearn.modelInfo (CAS Actions), 9
- cas.deepLearn.removeLayer (CAS Actions), 9
- cas.deepNeural.dnnCode (CAS Actions), 9
- cas.deepNeural.dnnExportModel (CAS Actions), 9
- cas.deepNeural.dnnScore (CAS Actions), 9
- cas.deepNeural.dnnTrain (CAS Actions), 9
- cas.deepRnn.rnnScore (CAS Actions), 9
- cas.deepRnn.rnnTrain (CAS Actions), 9
- cas.ds2.runDS2 (CAS Actions), 9
- cas.ds2.runModel (CAS Actions), 9
- cas.espCluster.listservers (CAS Actions), 9
- cas.espCluster.startservers (CAS Actions), 9
- cas.factmac.factmac (CAS Actions), 9
- cas.fastKnn.fastknn (CAS Actions), 9
- cas.FCMPACT.addRoutines (CAS Actions), 9
- cas.FCMPACT.loadFcmpLibs (CAS Actions), 9
- cas.FCMPACT.loadFcmpTable (CAS Actions), 9
- cas.FCMPACT.runProgram (CAS Actions), 9
- cas.fedSql.execDirect (CAS Actions), 9

- cas.freqTab.freqTab (CAS Actions), 9
- cas.gam.gampl (CAS Actions), 9
- cas.gam.gamScore (CAS Actions), 9
- cas.gVarCluster.gvarcluster (CAS Actions), 9
- cas.hiddenMarkovModel.hmm (CAS Actions), 9
- cas.hyperGroup.hypergroup (CAS Actions), 9
- cas.hyperGroup.thePlotThickens (CAS Actions), 9
- cas.image.augmentImages (CAS Actions), 9
- cas.image.compareImages (CAS Actions), 9
- cas.image.condenseImages (CAS Actions), 9
- cas.image.fetchImages (CAS Actions), 9
- cas.image.flattenImageTable (CAS Actions), 9
- cas.image.loadImages (CAS Actions), 9
- cas.image.matchImages (CAS Actions), 9
- cas.image.processImages (CAS Actions), 9
- cas.image.saveImages (CAS Actions), 9
- cas.image.summarizeImages (CAS Actions), 9
- cas.ldaTopic.ldaScore (CAS Actions), 9
- cas.ldaTopic.ldaTrain (CAS Actions), 9
- cas.loadStreams.appendSnapshot (CAS Actions), 9
- cas.loadStreams.loadSnapshot (CAS Actions), 9
- cas.loadStreams.loadStream (CAS Actions), 9
- cas.loadStreams.mMetaData (CAS Actions), 9
- cas.localSearch.solveLso (CAS Actions), 9
- cas.max, 14
- cas.mean, 15
- cas.median, 16
- cas.min, 17
- cas.mode, 18
- cas.modelPublishing.copyModelExternal (CAS Actions), 9
- cas.modelPublishing.deleteModel (CAS Actions), 9
- cas.modelPublishing.publishModel (CAS Actions), 9
- cas.modelPublishing.publishModelExternal (CAS Actions), 9
- cas.modelPublishing.runModelExternal (CAS Actions), 9
- cas.modelPublishing.runModelLocal (CAS Actions), 9
- cas.network.biconnectedComponents (CAS Actions), 9
- cas.network.centrality (CAS Actions), 9
- cas.network.clique (CAS Actions), 9
- cas.network.community (CAS Actions), 9
- cas.network.connectedComponents (CAS Actions), 9
- cas.network.core (CAS Actions), 9
- cas.network.cycle (CAS Actions), 9
- cas.network.path (CAS Actions), 9
- cas.network.reach (CAS Actions), 9
- cas.network.readGraph (CAS Actions), 9
- cas.network.shortestPath (CAS Actions), 9
- cas.network.summary (CAS Actions), 9
- cas.network.transitiveClosure (CAS Actions), 9
- cas.neuralNet.annCode (CAS Actions), 9
- cas.neuralNet.annScore (CAS Actions), 9
- cas.neuralNet.annTrain (CAS Actions), 9
- cas.nmiss, 18
- cas.optimization.convertMps (CAS Actions), 9
- cas.optimization.loadMps (CAS Actions), 9
- cas.optimization.solveLp (CAS Actions), 9
- cas.optimization.solveMilp (CAS Actions), 9
- cas.optimization.solveQp (CAS Actions), 9
- cas.optimization.tuner (CAS Actions), 9
- cas.optNetwork.biconnectedComponents (CAS Actions), 9
- cas.optNetwork.clique (CAS Actions), 9
- cas.optNetwork.connectedComponents (CAS Actions), 9
- cas.optNetwork.cycle (CAS Actions), 9
- cas.optNetwork.LAP (CAS Actions), 9
- cas.optNetwork.linearAssignment (CAS Actions), 9
- cas.optNetwork.MCF (CAS Actions), 9
- cas.optNetwork.minCostFlow (CAS

- Actions), 9
- cas.optNetwork.minCut (CAS Actions), 9
- cas.optNetwork.minSpanTree (CAS Actions), 9
- cas.optNetwork.MST (CAS Actions), 9
- cas.optNetwork.path (CAS Actions), 9
- cas.optNetwork.readGraph (CAS Actions), 9
- cas.optNetwork.shortestPath (CAS Actions), 9
- cas.optNetwork.summary (CAS Actions), 9
- cas.optNetwork.transitiveClosure (CAS Actions), 9
- cas.optNetwork.tsp (CAS Actions), 9
- cas.override.override (CAS Actions), 9
- cas.panel.panel (CAS Actions), 9
- cas.pca.eig (CAS Actions), 9
- cas.pca.itergs (CAS Actions), 9
- cas.pca.nipals (CAS Actions), 9
- cas.pca.randompca (CAS Actions), 9
- cas.percentile.assess (CAS Actions), 9
- cas.percentile.boxPlot (CAS Actions), 9
- cas.percentile.percentile (CAS Actions), 9
- cas.phreg.cox (CAS Actions), 9
- cas.pls.pls (CAS Actions), 9
- cas.probt, 19
- cas.qkb.importQKBFromCaslib (CAS Actions), 9
- cas.qkb.importQKBFromURL (CAS Actions), 9
- cas.qkb.listQKBDefinitions (CAS Actions), 9
- cas.qkb.listQKBLocales (CAS Actions), 9
- cas.qkb.listQKBs (CAS Actions), 9
- cas.qkb.listQKBTokens (CAS Actions), 9
- cas.qkb.loadQKB (CAS Actions), 9
- cas.qkb.removeQKB (CAS Actions), 9
- cas.qlim.qlim (CAS Actions), 9
- cas.quantile, 20
- cas.quantreg.quantreg (CAS Actions), 9
- cas.read.csv, 21, 23, 24, 27, 29, 31
- cas.read JMP, 22, 22, 24, 27, 29, 31
- cas.read SAS7BDAT, 22, 23, 24, 27, 29, 31
- cas.read table, 22–24, 25, 29, 31
- cas.read.xlsx, 22–24, 27, 28, 31
- cas.readRDS, 22–24, 27, 29, 30
- cas.recommend.recomAls (CAS Actions), 9
- cas.recommend.recomCreate (CAS Actions), 9
- cas.recommend.recomKnnScore (CAS Actions), 9
- cas.recommend.recomKnnTrain (CAS Actions), 9
- cas.recommend.recomMfScore (CAS Actions), 9
- cas.recommend.recomRateInfo (CAS Actions), 9
- cas.recommend.recomSample (CAS Actions), 9
- cas.recommend.recomSearchIndex (CAS Actions), 9
- cas.recommend.recomSearchQuery (CAS Actions), 9
- cas.recommend.recomSim (CAS Actions), 9
- cas.regression.genmod (CAS Actions), 9
- cas.regression.genmodScore (CAS Actions), 9
- cas.regression.glm (CAS Actions), 9
- cas.regression.glmScore (CAS Actions), 9
- cas.regression.logistic (CAS Actions), 9
- cas.regression.logisticScore (CAS Actions), 9
- cas.robustPca.mwpca (CAS Actions), 9
- cas.robustPca.robustpca (CAS Actions), 9
- cas.ruleMining.fism (CAS Actions), 9
- cas.ruleMining.mbanalysis (CAS Actions), 9
- cas.sampling.oversample (CAS Actions), 9
- cas.sampling.srs (CAS Actions), 9
- cas.sampling.stratified (CAS Actions), 9
- cas.sandcas.sand (CAS Actions), 9
- cas.saveRDS, 31, 40, 42, 43
- cas.sccas1.runCas1 (CAS Actions), 9
- cas.sd, 32
- cas.search.appendIndex (CAS Actions), 9
- cas.search.buildIndex (CAS Actions), 9
- cas.search.deleteDocuments (CAS Actions), 9
- cas.search.getSchema (CAS Actions), 9
- cas.search.searchAggregate (CAS Actions), 9
- cas.search.searchIndex (CAS Actions), 9
- cas.search.valuecount (CAS Actions), 9
- cas.searchAnalytics.buildAutoComplete (CAS Actions), 9

- cas.searchAnalytics.searchAutoComplete (CAS Actions), 9
- cas.sentimentAnalysis.applySent (CAS Actions), 9
- cas.sequence.pathing (CAS Actions), 9
- cas.session.addNodeStatus (CAS Actions), 9
- cas.session.batchresults (CAS Actions), 9
- cas.session.endSession (CAS Actions), 9
- cas.session.fetchresult (CAS Actions), 9
- cas.session.flushresult (CAS Actions), 9
- cas.session.listresults (CAS Actions), 9
- cas.session.listSessions (CAS Actions), 9
- cas.session.metrics (CAS Actions), 9
- cas.session.sessionId (CAS Actions), 9
- cas.session.sessionName (CAS Actions), 9
- cas.session.sessionStatus (CAS Actions), 9
- cas.session.setLocale (CAS Actions), 9
- cas.session.timeout (CAS Actions), 9
- cas.sessionProp.addFmtLib (CAS Actions), 9
- cas.sessionProp.addFormat (CAS Actions), 9
- cas.sessionProp.combineFmtLibs (CAS Actions), 9
- cas.sessionProp.deleteFormat (CAS Actions), 9
- cas.sessionProp.dropFmtLib (CAS Actions), 9
- cas.sessionProp.getSessOpt (CAS Actions), 9
- cas.sessionProp.listFmtLibs (CAS Actions), 9
- cas.sessionProp.listFmtRanges (CAS Actions), 9
- cas.sessionProp.listFmtSearch (CAS Actions), 9
- cas.sessionProp.listFmtValues (CAS Actions), 9
- cas.sessionProp.listSessOpts (CAS Actions), 9
- cas.sessionProp.promoteFmtLib (CAS Actions), 9
- cas.sessionProp.saveFmtLib (CAS Actions), 9
- cas.sessionProp.setFmtSearch (CAS Actions), 9
- cas.sessionProp.setSessOpt (CAS Actions), 9
- cas.severity.severity (CAS Actions), 9
- cas.severity.severityValidate (CAS Actions), 9
- cas.simple.correlation (CAS Actions), 9
- cas.simple.crossTab (CAS Actions), 9
- cas.simple.distinct (CAS Actions), 9
- cas.simple.freq (CAS Actions), 9
- cas.simple.groupBy (CAS Actions), 9
- cas.simple.mdSummary (CAS Actions), 9
- cas.simple.numRows (CAS Actions), 9
- cas.simple.paraCoord (CAS Actions), 9
- cas.simple.regression (CAS Actions), 9
- cas.simple.summary (CAS Actions), 9
- cas.simple.topK (CAS Actions), 9
- cas.simpleForecast.forecast (CAS Actions), 9
- cas.smartData.dataSegment (CAS Actions), 9
- cas.spatialreg.spatialreg (CAS Actions), 9
- cas.spc.boxChart (CAS Actions), 9
- cas.spc.cChart (CAS Actions), 9
- cas.spc.irChart (CAS Actions), 9
- cas.spc.mChart (CAS Actions), 9
- cas.spc.mrChart (CAS Actions), 9
- cas.spc.npChart (CAS Actions), 9
- cas.spc.pChart (CAS Actions), 9
- cas.spc.rChart (CAS Actions), 9
- cas.spc.sChart (CAS Actions), 9
- cas.spc.uChart (CAS Actions), 9
- cas.spc.xChart (CAS Actions), 9
- cas.spc.xrChart (CAS Actions), 9
- cas.spc.xsChart (CAS Actions), 9
- cas.stabilityMonitoring.smCalib (CAS Actions), 9
- cas.stabilityMonitoring.smScore (CAS Actions), 9
- cas.stderr, 33
- cas.sum, 34
- cas.svDataDescription.svddTrain (CAS Actions), 9
- cas.svm.svmTrain (CAS Actions), 9
- cas.table.addCaslib (CAS Actions), 9
- cas.table.addTable (CAS Actions), 9

- cas.table.alterTable (CAS Actions), 9
- cas.table.attribute (CAS Actions), 9
- cas.table.caslibInfo (CAS Actions), 9
- cas.table.columnInfo (CAS Actions), 9
- cas.table.deleteSource (CAS Actions), 9
- cas.table.dropCaslib (CAS Actions), 9
- cas.table.dropTable (CAS Actions), 9
- cas.table.fetch (CAS Actions), 9
- cas.table.fileInfo (CAS Actions), 9
- cas.table.index (CAS Actions), 9
- cas.table.loadDataSource (CAS Actions), 9
- cas.table.loadTable (CAS Actions), 9
- cas.table.partition (CAS Actions), 9
- cas.table.promote (CAS Actions), 9
- cas.table.queryCaslib (CAS Actions), 9
- cas.table.recordCount (CAS Actions), 9
- cas.table.save (CAS Actions), 9
- cas.table.shuffle (CAS Actions), 9
- cas.table.tableDetails (CAS Actions), 9
- cas.table.tableExists (CAS Actions), 9
- cas.table.tableInfo (CAS Actions), 9
- cas.table.update (CAS Actions), 9
- cas.table.upload (CAS Actions), 9
- cas.table.view (CAS Actions), 9
- cas.terminate, 35
- cas.textMining.tmMine (CAS Actions), 9
- cas.textMining.tmScore (CAS Actions), 9
- cas.textMining.tmSvd (CAS Actions), 9
- cas.textParse.tpAccumulate (CAS Actions), 9
- cas.textParse.tpParse (CAS Actions), 9
- cas.textRuleDevelop.compileCategory (CAS Actions), 9
- cas.textRuleDevelop.compileConcept (CAS Actions), 9
- cas.textRuleDevelop.validateCategory (CAS Actions), 9
- cas.textRuleDevelop.validateConcept (CAS Actions), 9
- cas.textRuleDiscover.termMap (CAS Actions), 9
- cas.textRuleScore.applyCategory (CAS Actions), 9
- cas.textRuleScore.applyConcept (CAS Actions), 9
- cas.textRuleScore.loadTableFromDisk (CAS Actions), 9
- cas.textSummarization.textSummarize (CAS Actions), 9
- cas.textTopic.tmCreateTopic (CAS Actions), 9
- cas.textTopic.tmMergeTopic (CAS Actions), 9
- cas.textUtil.tmAstore (CAS Actions), 9
- cas.textUtil.tmFindSimilar (CAS Actions), 9
- cas.timeData.forecast (CAS Actions), 9
- cas.timeData.runTimeCode (CAS Actions), 9
- cas.timeData.timeSeries (CAS Actions), 9
- cas.timeFilters.expectedRange (CAS Actions), 9
- cas.timeFrequency.stft (CAS Actions), 9
- cas.timeFrequency.window (CAS Actions), 9
- cas.transpose.transpose (CAS Actions), 9
- cas.tsInfo.getInfo (CAS Actions), 9
- cas.tsInfo.selectLag (CAS Actions), 9
- cas.tsReconcile.reconcileTwoLevels (CAS Actions), 9
- cas.tvalue, 35
- cas.upload, 36
- cas.upload.file, 36
- cas.upload.frame, 37
- cas.uss, 37
- cas.var, 38
- cas.varReduce.super (CAS Actions), 9
- cas.varReduce.unsuper (CAS Actions), 9
- cas.write.csv, 32, 39, 40, 42, 43
- cas.write.csv2, 40
- cas.write.table, 32, 40, 41, 43
- cas.write.xlsx, 32, 40, 42, 42
- casDataFrame, 54, 67, 68, 71
- casDataFrame (casDataFrame-class), 43
- casDataFrame-class, 43
- CASTable, 8, 9, 22–24, 27, 29, 31, 36, 37, 46, 47, 50–55, 60–64, 66–68, 71
- CASTable (CASTable-class), 44
- CASTable-class, 44
- cbind.casTable, 45
- cbind2, CASTable, ANY-method (cbind2.casTable), 45
- cbind2.casTable, 45
- colMeans, CASTable-method, 46
- colNames, CASTable-method, 47

colSums, CASTable-method, 48
cor, CASTable-method, 48
cov, CASTable-method, 49

defCasTable, 50
dim, CASTable-method, 51
dimnames, CASTable-method, 52
download_sas_binaries, 52
dropTable, 53

head, CASTable-method, 54

is.castable, 54

length, CASTable-method, 55
listActionParms, 56
listActionSets, 56
loadActionSet, 57

max, CASTable-method, 58
mean, CASTable-method, 58
median, CASTable-method, 59
min, CASTable-method, 60

names, CASTable-method, 60
ncol, CASTable-method, 61
nrow, CASTable-method, 62

rbind, CASTable-method, 62
rbind.casTable, 63
rbind2, CASTable, ANY-method
 (rbind2.casTable), 63
rbind2.casTable, 63
rownames, CASTable-method, 64
runAction, 64

show, casDataFrame-method, 65
subset.casTable, 66
summary, CASTable-method, 66
swat, 67
swat-package (swat), 67

tail, CASTable-method, 68
to.casDataFrame, 69
to.data.frame, 70
to.r.data.frame, 70

unique, CASTable-method, 71