

Package ‘colbycol’

January 2, 2012

Type Package

Title Read big text files column by column, sometimes much bigger than available RAM, into R.

Version 0.7

Date 2011-04-17

Author Carlos J. Gil Bellosta

Maintainer Carlos J. Gil Bellosta <cgb@datanalytics.com>

Description This package tries to solve the memory restrictions posed by large text files by breaking them into their columns first. These columns are later read individually into R. The package also provides some helper functions for manipulating the newly created objects.

URL <http://colbycol.r-forge.r-project.org>

Depends rJava, filehash

License GPL

LazyLoad yes

Repository CRAN

Repository/R-Forge/Project colbycol

Repository/R-Forge/Revision 23

Date/Publication 2011-05-23 15:58:58

R topics documented:

colbycol-package	2
as.data.frame.colbycol	2
cbc.get.col	3
cbc.read.table	4
cbc.save	6
cbc.test.data	7
colnames.colbycol	7
ncol.colbycol	8

Index**10**

colbycol-package	<i>Reads data text files in a memory efficient way</i>
------------------	--

Description

This package tries to solve the memory restrictions posed by large text files by breaking them into its columns, which are subsequently read individually into R.

Details

Package:	colbycol
Type:	Package
Version:	0.4
Date:	2009-07-29
License:	GPL
LazyLoad:	yes

Author(s)

Maintainer: Carlos J. Gil Bellosta <cgb@datanalytics.com>

as.data.frame.colbycol	<i>Creates a data.frame from a colbycol object after possibly filtering either rows or columns</i>
------------------------	--

Description

Function `as.data.frame` transforms a colbycol object into a regular data.frame object after possibly filtering out some columns or rows.

Usage

```
## S3 method for class 'colbycol'  
as.data.frame( x, row.names, optional, ..., rows = NULL, columns = NULL )
```

Arguments

x	a colbycol object
row.names	unimplemented
optional	unimplemented
columns	either a character or a numeric vector indicating the selected columns from x which the data.frame will consist of; the default value NULL will select them all
rows	a numeric vector indicating the selected rows from x which the data.frame will consist of; the default value NULL will select them all
...	unimplemented

Details

After a colbycol object is created, it is sometimes possible to transform it into a data.frame object stored in RAM. This function can create a data.frame object selecting just a number of rows and columns from the colbycol object.

Value

A data.frame with the selected columns and rows.

Author(s)

Carlos J. Gil Bellosta

Examples

```
cbc.data <- cbc.read.table( system.file("extdata",
  "cbc.test.data.txt", package = "colbycol"), sep = "\t" )

my.df <- as.data.frame( cbc.data )
my.df <- as.data.frame( cbc.data, columns = 2:4, rows = 20:60 )
```

```
cbc.get.col
```

Reads a single column from the original file into memory

Description

Function `cbc.read.table` reads a file, stores it column by column in disk file and creates a colbycol object. Function `cbc.get.col` queries this object and returns a single column.

Usage

```
cbc.get.col(data, column)
```

Arguments

data a colbycol object
column a single number of column name indicating the column to be retrieved from disk

Details

column can be either an existing column name from the file (see `colnames.colbycol`) or a number indicating the correlative order of the column within data.

Value

The data contained in the indicated column in the original file.

Author(s)

Carlos J. Gil Bellosta

Examples

```
cbc.data <- cbc.read.table( system.file("extdata",  
  "cbc.test.data.txt", package = "colbycol"), sep = "\t" )  
ncol( cbc.data )  
colnames( cbc.data )  
col.01 <- cbc.get.col( cbc.data, 1)  
col.02 <- cbc.get.col( cbc.data, "col02" )
```

cbc.read.table *Reads a huge text file and creates a colbycol object*

Description

cbc.read.table is able to read a huge text data file well beyond the memory restrictions imposed by `read.table`. It reads the file line by line, breaks it into as many physical files as columns, reads them back into R one by one and saves them in efficient, native R data files.

Usage

```
cbc.read.table(file, filehash.name = tempfile( pattern = "filehash_" ), tmp.dir = tempfile( pattern = "  
just.read = NULL, sample.pct = NULL, sep = "\t",  
header = TRUE, ...)
```

Arguments

<code>file</code>	file to be loaded
<code>filehash.name</code>	name of the filehash where data will be stored after being read; it will be created via the tools in package <code>filehash</code>
<code>tmp.dir</code>	path to the (empty) directory where temporary files are stored
<code>just.read</code>	vector of column names, numeric positions, or logical pattern for those columns in the file that need to be read; the default value <code>all</code> loads all columns
<code>sample.pct</code>	either <code>NULL</code> (no sampling) or a value (estricly) between 0 and 1 indicating the proportion of rows to read from <code>file</code>
<code>sep</code>	field separator for the data in <code>file</code>
<code>header</code>	whether <code>file</code> contains headers or not
<code>...</code>	other parameters passed to <code>read.table</code> internally

Details

This function invokes a python script which reads `file` line by line, breaks each of them into tokens as indicated by `sep`, and stores each column in an independent text file in `tmp.dir`. These files are then read into R one by one and the text files are replaced by R native data files stored with `save`. The function returns a tiny object containing the required metadata, while the data sits in `tmp.dir`.

It is possible both to skip a number of rows passing the `skip` parameter to the inner functions. It is also possible to read just a subset of columns by identifying them via the `just.read` parameter or to sample a given proportion of the rows in `file` providing a number between 0 and 1 via the `sample.pct` parameter. Note that if after filtering out a subset of rows or columns there are none available, the function fails.

It is convenient to provide the full path to `tmp.dir` in case the working directory is modified; otherwise, the temporary files could not be found. If no temporary directory is provided, a temporal one is created that will be erased as the R session ends.

Caution is required to pass extra arguments to the internal calls to `read.table` via `...`

Value

An object of class `colbycol` containing the metadata required to access the data from the original file that is stored in `tmp.dir`.

Author(s)

Carlos J. Gil Bellosta

See Also

[read.table](#), [save](#)

Examples

```
cbc.data <- cbc.read.table( system.file("extdata", "cbc.test.data.txt",
  package = "colbycol"), sep = "\t" )
nrow( cbc.data )
colnames( cbc.data )
col.01 <- cbc.get.col( cbc.data, 1)
col.02 <- cbc.get.col( cbc.data, "col02" )

cbc.data.sampled <- cbc.read.table( system.file("extdata",
  "cbc.test.data.txt", package = "colbycol"),
  sample.pct = 0.4, sep = "\t" )
cbc.data.2.cols <- cbc.read.table( system.file("extdata",
  "cbc.test.data.txt", package = "colbycol"),
  just.read = c( 1, 3), sep = "\t" )
```

cbc.save

Save and load colbycol objects saved on disk

Description

These functions save and load colbycol objects, possibly for storage between sessions.

Usage

```
cbc.save( data )
cbc.load( file.name )
```

Arguments

data	a colbycol object
file.name	path to the filehash object to be loaded

Value

Function `cbc.load`, if successful, returns an object of class `colbycol`. Function `cbc.save`, for convenience, returns the name of the file where data has been stored.

Author(s)

Carlos J. Gil Bellosta

Examples

```
cbc.data <- cbc.read.table( system.file("extdata",
  "cbc.test.data.txt", package = "colbycol"), sep = "\t" )
new.cbc.data <- cbc.load( cbc.save( cbc.data ) )
```

cbc.test.data	<i>Simple test data</i>
---------------	-------------------------

Description

The testdata dataset is a small dataset to test the colnames package. Its contents are, otherwise, meaningless.

Usage

```
cbc.test.data
```

Format

This data frame contains the following columns:

col01 numeric

col02 numeric

col03 numeric

col04 text

col05 text

Source

Randomly generated.

Examples

```
data( cbc.test.data )
head( cbc.test.data )
```

colnames.colbycol	<i>Lists the column names in a colbycol object</i>
-------------------	--

Description

colnames lists the column names in a colbycol object. It is an extension of the [colnames](#) function.

Usage

```
## S3 method for class 'colbycol'
colnames(x, do.NULL, prefix)
```

Arguments

x	a colbycol object
do.NULL	required for the colnames default function
prefix	required for the colnames default function

Value

A character vector containing the names of the columns in x.

Author(s)

Carlos J. Gil Bellosta

See Also

[colnames](#)

Examples

```
cbc.data <- cbc.read.table( system.file("extdata",
  "cbc.test.data.txt", package = "colbycol"), sep = "\t" )
colnames( cbc.data )
```

ncol.colbycol

Returns the number of rows/cols in a colbycol object

Description

This function extends the [ncol](#) function for regular data.frames.

Usage

```
## S3 method for class 'colbycol'
ncol(x)
## S3 method for class 'colbycol'
nrow(x)
## S3 method for class 'colbycol'
summary( object, ...)
```

Arguments

x	a colbycol object
object	a colbycol object
...	currently unused

Value

An integer.

Author(s)

Carlos J. Gil Bellosta

See Also

[ncol](#), [nrow](#)

Examples

```
cbc.data <- cbc.read.table( system.file("extdata",  
  "cbc.test.data.txt", package = "colbycol"), sep = "\t" )  
nrow( cbc.data )  
ncol( cbc.data )  
summary( cbc.data )
```

Index

*Topic **datasets**

cbc.test.data, 7

*Topic **manip**

as.data.frame.colbycol, 2

cbc.get.col, 3

cbc.read.table, 4

cbc.save, 6

colbycol-package, 2

colnames.colbycol, 7

ncol.colbycol, 8

*Topic **package**

colbycol-package, 2

as.data.frame, 2

as.data.frame(as.data.frame.colbycol),

2

as.data.frame.colbycol, 2

cbc.get.col, 3, 3

cbc.load(cbc.save), 6

cbc.read.table, 3, 4

cbc.save, 6

cbc.test.data, 7

colbycol(colbycol-package), 2

colbycol-package, 2

colnames, 7, 8

colnames(colnames.colbycol), 7

colnames.colbycol, 4, 7

ncol, 8, 9

ncol(ncol.colbycol), 8

ncol.colbycol, 8

nrow, 9

nrow(ncol.colbycol), 8

read.table, 4, 5

save, 5

summary.colbycol(ncol.colbycol), 8