Package ‘scidb’

April 15, 2017

Type Package
Title An R Interface to SciDB
Version 2.0.0
Date 2017-4-14
Copyright Paradigm4, Inc.
BugReports https://github.com/Paradigm4/SciDBR/issues
URL http://paradigm4.github.io/SciDBR
VignetteBuilder knitr
Depends R (>= 3.0.0), bit64
Imports curl, data.table, digest, methods, openssl
Suggests Matrix, knitr (>= 1.8)
License AGPL-3
LazyLoad yes
RoxygenNote 6.0.1.9000
NeedsCompilation yes
Author B. W. Lewis [cre, aut],
Alex Poliakov [aut]
Maintainer B. W. Lewis <blewis@illposed.net>
Repository CRAN
Date/Publication 2017-04-14 23:05:25 UTC

R topics documented:

scidb-package .......................................................... 2
aflhelp ................................................................. 3
as.R ................................................................. 3
as.scidb ............................................................. 5
at_least ............................................................... 6
Description

SciDB/R Interface

Package options

options(scidb.prefix=NULL) # Default shim port and host. options(scidb.default_shim_port=8080L)
options(scidb.default_shim_host="localhost") # How to download arrays and their coordinates. Set
scidb.unpack=FALSE # to use apply, which can be faster in some cases when used with aio. op-
tions(scidb.unpack=FALSE) # Disable SSL certificate host name checking by default. This is im-
portant mostly # for Amazon EC2 where hostnames rarely match their DNS names. If you enable
this # then the shim SSL certificate CN entry *must* match the server host name for the # encrypted
session to work. Set this TRUE for stronger security (help avoid MTM) # in SSL connections. op-
tions(scidb.verifyhost=FALSE) # List of special DDL operators options(scidb.ddl=c("create_array",
"remove", "rename"))

See Also

scidb, iquery
**aflhelp**

Display SciDB AFL operator documentation

**Description**
Display SciDB AFL operator documentation

**Usage**
aflhelp(topic, db)

**Arguments**
- **topic**: an AFL object from a SciDB database connection, or optionally a character string name
- **db**: optional database connection from scidbconnect (only needed when topic is a character string)

**Value**
displays help

**Examples**
```r
## Not run:
d <- scidbconnect()
aflhelp("list", d)  # explicitly look up a character string
help(d$list)       # same thing via R's \code{help} function
## End(Not run)
```

---

**as.R**

Download SciDB data to R

**Description**
Download SciDB data to R

**Usage**
```r
as.R(x, only_attributes = FALSE, binary = TRUE)
```
Arguments

- **x**: a `scidb` object (a SciDB array or expression)
- **only_attributes**: optional logical argument, if `TRUE` do not download SciDB dimensions (see note)
- **binary**: optional logical value, set to `FALSE` to download data using text format (useful for some unsupported SciDB types)

Value

An R `data.frame`

Note

This convenience function is equivalent to running `iquery(db, x, return=TRUE)` for a SciDB connection object `db`.

The `only_attributes=TRUE` option only works with binary transfers, and if specified will set `binary=TRUE`. Beware of the `only_attributes=TRUE` setting—SciDB may return data in arbitrary order.

SciDB values are always returned as R data frames. SciDB scalar types are converted to corresponding R types as follows:

- `double` -> `double`
- `int64` -> `integer64`
- `uint64` -> `double`
- `uint32` -> `double`
- `int32` -> `integer`
- `int16` -> `integer`
- `uint16` -> `integer`
- `int8` -> `integer`
- `uint8` -> `integer`
- `bool` -> `logical`
- `string` -> `character`
- `char` -> `character`
- `binary` -> `raw`
- `datetime` -> `Date`

See Also

`as.scidb`
Examples

## Not run:

db <- scidbconnect()
x <- scidb(db, "build(<v:double>[i=1:5], sin(i))")
as.R(x)

## i

## 1 0.8414710
## 2 0.9092974
## 3 0.1411200
## 4 -0.7568025
## 5 -0.9589243

as.R(x, only_attributes=TRUE)

## v

## 0.8414710
## 0.9092974
## 0.1411200
## -0.7568025
## -0.9589243

## End(Not run)

---

**as.scidb**

*Upload R data to SciDB*

Description

Upload R data to SciDB

Usage

```r
as.scidb(db, x, name, start, gc = TRUE, ...)
```

Arguments

- **db**: a scidb database connection returned from `scidbconnect`
- **x**: an R data frame, raw value, Matrix, matrix, or vector object
- **name**: a SciDB array name to use
- **start**: starting SciDB integer coordinate index (does not apply to data frames)
- **gc**: set to FALSE to disconnect the SciDB array from R’s garbage collector
- **...**: other options, see `df2scidb`

Value

A scidb object
Note

Supported R objects include data frames, scalars, vectors, dense matrices, and double-precision sparse matrices of class CsparseMatrix. Supported R scalar types and their resulting SciDB types are:

- integer -> int32
- logical -> int32
- character -> string
- double -> double
- integer64 -> int64
- raw -> binary
- Date -> datetime

R factor values are converted to their corresponding character levels.

See Also

as.R

at_least(x, y)  

Returns TRUE if version string x is greater than or equal to version y

Arguments

x  
version string like "12.1", "15.12", etc. (non-numeric ignored)
y  
version string like "12.1", "15.12", etc. (non-numeric ignored)

Value

logical TRUE if x is greater than or equal to y
Covariance matrix

This function is more limited than R’s default `cov` function. It can only compute a covariance matrix from a data matrix without any missing value handling by the procedure (in R notation)

\[ S_0 \leftarrow \text{sweep}(x, 2, \text{colMeans}(x)). \text{crossprod}(S_0)/(\text{nrow}(S_0) - 1) \]  # (covariance matrix result)

### Usage

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

### Arguments

- `x`: a 2-d scidb array with a single numeric attribute
- `y`: UNUSED, limited to correlation matrix in the SciDB case
- `use`: UNUSED, limited to "everything" in the SciDB case
- `method`: UNUSED, limited to "pearson" in the SciDB case

#### Value

covariance matrix of `x` (as a SciDB array)

### dimnames.scidb

Names of array dimensions

#### Description

Names of array dimensions

#### Usage

```r
## S3 method for class 'scidb'
dimnames(x)
```

#### Arguments

- `x`: scidb array object
help,operator-method

Value

a vector of SciDB array dimension names

cgetpwd

Simple utility to interactively enter a password without showing it on the screen

Description

Simple utility to interactively enter a password without showing it on the screen

Usage

cgetpwd(prompt = "Password:"

Arguments

prompt a text prompt to display, defaults to "Password:"

help,operator-method AFL.operator help

Description

AFL operator help

Usage

## S4 method for signature 'operator'
help(topic)

Arguments

topic afl operator

Value

help summary
iquery

Run a SciDB query, optionally returning the result.

Description

Run a SciDB query, optionally returning the result.

Usage

iquery(db, query, return = FALSE, binary = TRUE, ...)

Arguments

db  a scidb database connection from scidbconnect
query a single SciDB query string or scidb array object
return if TRUE, return the result
binary set to FALSE to read result from SciDB in text form
... additional options passed to read.table when binary=FALSE, or optional result schema when binary=TRUE (see note below).

Note

When query is an arbitrary AFL query string and binary=TRUE, optionally specify schema with a valid result array schema to skip an extra metadata lookup query (see scidb).

Setting return=TRUE wraps the AFL query expression with a SciDB save operator, saving the data on the SciDB server in either binary or text format depending on the value of the binary parameter. Please note that some AFL expressions may not be "saved" using the AFL save operator, including for instance the AFL create_array operator. Trying to return the result of such a SciDB expression will result in a run-time error.

See Also

scidb as.R

Examples

```r
## Not run:
db <- scidbconnect()
iquery(db, "build(v:double)[i=1:5], sin(i)"", return=TRUE)
## i
## v
## 1 0.8414710
## 2 0.9092974
## 3 0.1411200
## 4 -0.7568025
## 5 -0.9589243

# Use binary=FALSE and additional options to read.table function:
```
iquery(db, "build<val:string>[i=1:3], '[(%(01),(02),(03)]', true",xi
   return=TRUE, binary=FALSE, colclasses=c("integer", "character"))

## i val
## 1 1 01
## 2 2 02
## 3 3 03

## End(Not run)

---

### `ls,afl-method`

**List contents of a SciDB database**

**Description**

List contents of a SciDB database

**Usage**

```r
## S4 method for signature 'afl'
ls(name)
```

**Arguments**

- `name` : `afl` SciDB connection object from `scidbconnect`

**Value**

- A `data.frame` listing the contents of the database

---

### `names.scidb`

**SciDB dimension and attribute names**

**Description**

SciDB dimension and attribute names

**Usage**

```r
## S3 method for class 'scidb'
names(x)
```

**Arguments**

- `x` : `scidb` array object

**Value**

- Character vector of names
operators

<table>
<thead>
<tr>
<th>operators</th>
<th>Base SciDB operators</th>
</tr>
</thead>
</table>

**Description**

Base SciDB operators as of SciDB version 16.9

**Usage**

data(operators)

**Format**

A data frame with 4 variables, name, signature, help.

**Source**

Paradigm4 [http://paradigm4.com](http://paradigm4.com)

---

**print.afl-method**  
Print a summary of a afl SciDB database connection object

**Description**

Print a summary of a afl SciDB database connection object

**Usage**

```r
## S4 method for signature 'afl'
print(x)
```

**Arguments**

- `x`  
afl object

**Value**

printed object summary
**print.scidb-method**  
*Print a summary of a scidb object*

---

**Description**

Print a summary of a scidb object

**Usage**

```r
## S4 method for signature 'scidb'
print(x)
```

**Arguments**

- `x` scidb object

**Value**

printed object summary

---

**print.afl**  
*Print a summary of a afl SciDB database connection object*

---

**Description**

Print a summary of a afl SciDB database connection object

**Usage**

```r
## S3 method for class 'afl'
print(x, ...)
```

**Arguments**

- `x` afl object
- `...` optional arguments (not used)

**Value**

printed object summary
schema

SciDB array schema

Description

SciDB array schema

Usage

```
schema(x, what = c("schema", "attributes", "dimensions"))
```

Arguments

- **x**: a `scidb` array object
- **what**: optional schema subset (subsets are returned in data frames; partial argument matching is supported)

Value

character-valued SciDB array schema

Examples

```r
## Not run:
s <- scidbconnect()
x <- scidb(s,"build(<v:double>[i=1:10,2,0,j=0:19,1,0],0)"
schema(x)
# [1] "<v:double> [i=1:10:0:2; j=0:19:0:1]"
schema(x, "attributes")
# name type nullable
#1 v double TRUE
schema(x, "dimensions")
  name start end chunk overlap
#1  i  1  10  2  j
#2  0  0  19  1  0
## End(Not run)
```

scidb

Create an R reference to a SciDB array or expression.

Description

Create an R reference to a SciDB array or expression.
Usage

scidb(db, name, gc = FALSE, schema)

Arguments

db scidb connection object from scidbconnect
name a character string name of a stored SciDB array or a valid SciDB AFL expression
gc a logical value, TRUE means connect the SciDB array to R’s garbage collector
schema optional SciDB array schema, if specified avoid an extra metadata query to determine array schema. Use this option with care, the schema must exactly match the SciDB array result.

Value

a scidb object

scidbconnect Connect to a SciDB database

Description

Connect to a SciDB database

Usage

scidbconnect(host = getOption("scidb.default_shim_host", "127.0.0.1"),
port = getOption("scidb.default_shim_port", 8080L), username, password,
auth_type = c("scidb", "digest"), protocol = c("http", "https"),
int64 = FALSE, doc)

Arguments

host optional host name or I.P. address of a SciDB shim service to connect to
port optional port number of a SciDB shim service to connect to
username optional authentication username
password optional authentication password
auth_type optional SciDB authentication type
protocol optional shim protocol type
int64 logical value, if TRUE then preserve signed 64-bit SciDB integers as R integer64 values from the bit64 package. Otherwise, 64-bit integers from SciDB are converted to R double values, possibly losing precision.
doc optional AFL operator/macro documentation (see notes)
Value

A scidb connection object. Use `$` to access AFL operators and macros, `ls()` on the returned object to list SciDB arrays, and `names()` on the returned object to list all available AFL operators and macros.

Note

Use the optional `username` and `password` arguments with `auth_type` set to "digest" to use HTTP digest authentication (see the shim documentation to configure this). Digest authentication may use either "http" or "https" selected by the `protocol` setting. Set `auth_type = "scidb"` to use SciDB authentication, which only works over "https".

Use the returned SciDB connection object (of class `afl`) with other package functions to interact with SciDB arrays. Apply R’s `ls` function on the returned value to see a list of arrays. The returned value contains a list of available SciDB AFL language operators and macro names. Use the dollar-sign function to access those functions.

The optional `doc` argument may be a three-column data frame with character-valued columns `name`, `signature`, and `help` containing AFL operator names, function signatures, and help strings, respectively. See `data("operators", package="scidb")` for an example.

All arguments support partial matching.

See Also

scidb_prefix

Examples

```r
## Not run:
db <- scidbconnect()

# SciDB 15.12 authentication example (using shim's default HTTPS port 8083)
db <- scidbconnect(user="root", password="Paradigm4",
                   auth_type="scidb", port=8083, protocol="https")

# List available AFL operators
names(db)

# List arrays
ls(db)

# Explicitly upload an R matrix to SciDB:
x <- as.scidb(db, matrix(rnorm(20), 5))
# Implicitly do the same as part of an AFL expression
y <- db$join(x, as.scidb(matrix(1:20, 5)))
print(y)

as.R(y) # Download a SciDB array to R.

## End(Not run)
```
Description

SciDB AFL statements are normally executed in a stateless query context. Use `scidb_prefix` to create compound AFL expressions useful in some circumstances.

Usage

```
scidb_prefix(db, expression = NULL)
```

Arguments

- `db` - a scidb database connection returned from `scidbconnect`
- `expression` - a valid AFL expression to be issued prior to, and in the same context as all subsequent query expressions issued to the database corresponding to `db`. Set `expression=NULL` to remove the prefix expression.

Value

A new SciDB database connection object with the prefix set.

Note

This is mostly useful for setting namespaces, see the examples.

Examples

```r
## Not run:
library(scidb)
db <- scidbconnect()
ls(db)
new_db <- scidb_prefix(db, "set_role('functionary')")
ls(new_db)
## End(Not run)
```
**show,afl-method**  
*Print a summary of an afl object*

**Description**

Print a summary of an afl object

**Usage**

```r
## S4 method for signature 'afl'
show(object)
```

**Arguments**

- `object`: afl object

**Value**

printed object summary

---

**show,scidb-method**  
*Print a summary of a scidb object*

**Description**

Print a summary of a scidb object

**Usage**

```r
## S4 method for signature 'scidb'
show(object)
```

**Arguments**

- `object`: a scidb object

**Value**

printed object summary
store

*Evaluate an expression to scidb or scidb objects*

**Description**

Force evaluation of an expression that yields a scidb or scidb object, storing the result to a SciDB array when eval=TRUE.

**Usage**

```r
store(db, expr, name, eval = TRUE, gc = TRUE, temp = FALSE)
```

**Arguments**

- `db`: scidb connection object from `scidbconnect`
- `expr`: a quoted SciDB expression scidb object
- `name`: (character) optional SciDB array name to store result to
- `eval`: FALSE do not evaluate the expression in SciDB (leave as a view)
- `gc`: (logical) optional, when TRUE tie result to R garbage collector
- `temp`: (logical, optional), when TRUE store as a SciDB temp array

---

%as%,scidb-method

*AFL array aliasing*

**Description**

AFL array aliasing

**Usage**

```r
## S4 method for signature 'scidb'
x %as% y
```

**Arguments**

- `x`: an object of class `scidb` (a scidb array or expression)
- `y`: alias name

**Value**

A `scidb` object

**Note**

Use the `%as%` operator in place of the native AFL "as" operator in AFL expressions written in R.
Examples

```r
## Not run:
db <- scidbconnect()
x <- scidb(db, "build(<v:double>[i=1:2,1,0], i")
x %as% y

## End(Not run)
```

### Description

AFL array aliasing

### Usage

```r
x %as% y
```

### Arguments

- **x**: an object of class `scidb` (a scidb array or expression)
- **y**: alias name
Index

*Topic datasets
  operators, 11
  %as%, 19
  %as%, scidb-method, 18

afl, 3
aflhelp, 3
as.R, 3, 6, 9
as.scidb, 4, 5
at_least, 6

cov, scidb-method, 7
data.frame, 4
df2scidb, 5
dimnames.scidb, 7

getpwd, 8
help, operator-method, 8
iquery, 2, 9

ls, 15
ls, afl-method, 10

names.scidb, 10

operators, 11

print, afl-method, 11
print, scidb-method, 12
print.afl, 12

schema, 13
scidb, 2, 4, 9, 13, 18, 19
scidb-package, 2
scidb_prefix, 15, 16
scidbconnect, 3, 5, 9, 10, 14, 16, 18
show, afl-method, 17
show, scidb-method, 17
store, 18