

Package ‘hdfqlr’

July 22, 2025

Title Interface to 'HDFq1' API

Version 0.6-2

Description Provides an interface to 'HDFq1' <<https://www.hdfq1.com/>> and helper functions for reading data from and writing data to 'HDF5' files. 'HDFq1' provides a high-level language for managing 'HDF5' data that is platform independent. For more information, see the reference manual <<https://www.hdfq1.com/resources/HDFq1ReferenceManual.pdf>>.

Depends R (>= 3.4)

Imports utils, methods

Suggests bit64 (>= 0.9), knitr (>= 1.22), ggplot2 (>= 3.2), microbenchmark (>= 1.4), testthat (>= 2.1.0), rmarkdown (>= 2.8)

SystemRequirements HDFq1 (>= 2.1.0)

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.1.1

VignetteBuilder knitr

NeedsCompilation no

Author Michael Koohafkan [aut, cre]

Maintainer Michael Koohafkan <michael.koohafkan@gmail.com>

Repository CRAN

Date/Publication 2021-06-11 04:50:02 UTC

Contents

hdfqlr	2
hql	2
hql_create	3
hql_drop	4
hql_file	4
hql_flush	5

hql_is_loaded	5
hql_list	6
hql_load	6
hql_write	7

Index	9
--------------	----------

hdfqlr	<i>hdfqlr: an HDF API based on HDFql</i>
--------	--

Description

This package provides an R interface to HDF files using the **HDFql**.

Package options

hdfqlr uses the following `options()` to configure behavior:

- `hdfqlr.dir`: The HDFql install directory.

Alternatively, the HDFql install directory can be saved to an environment variable `HDFQL_DIR`.

Author(s)

Maintainer: Michael Koohafkan <michael.koohafkan@gmail.com>

hql	<i>HDFql Wrapper Constants and Functions</i>
-----	--

Description

Access the constants and functions provided by the HDFql wrapper. The wrapper contents are stored in an environment when the HDFql library is loaded and used internally by hdfqlr to perform operations.

Usage

hql

Format

An object of class environment of length 1.

Details

This environment is exported so that users can directly use the HDFql wrapper functions. The intended method of use is to `attach()` the environment to the search path. For more information on what is provided by the wrapper, consult the **HDFql reference manual**.

Examples

```
## Not run:
  attach(hql$wrapper)

## End(Not run)
```

hql_create

*Create HDF Files, Datasets or Attribute***Description**

Create HDF files and groups.

Usage

```
hql_create_file(file, overwrite = FALSE, parallel = FALSE)

hql_create_group(group, overwrite = FALSE)
```

Arguments

file	The HDF file to create.
overwrite	If TRUE, overwrite existing file, group, attribute, or dataset.
parallel	If TRUE, use parallel capabilities.
group	The group to create.

Functions

- hql_create_file: Create HDF file.
- hql_create_group: Create HDF group.

Examples

```
if(hql_is_loaded()) {
  tf = tempfile(fileext = ".h5")
  hql_create_file(tf)

  hql_use_file(tf)
  hql_create_group("group1")

  hql_close_file(tf)
}
```

hql_drop	<i>Drop HDF groups, datasets, and attributes</i>
----------	--

Description

Drop a dataset, attribute, or group from an HDF file.

Examples

```
if(hql_is_loaded()){
  tf = tempfile(fileext = ".h5")
  hql_create_file(tf)

  hql_use_file(tf)
  x = rnorm(10)
  attr(x, "myattribute") = "some information"
  hql_write_dataset(x, "mygroup/mydataset")

  hql_drop_attribute("mygroup/mydataset/myattribute")
  hql_drop_dataset("mygroup/mydataset")
  hql_drop_group("mygroup")

  hql_close_file(tf)
}
```

hql_file	<i>Access HDF Files</i>
----------	-------------------------

Description

Open (use) and close HDF files.

Usage

```
hql_use_file(file)

hql_close_file(file, all = FALSE)
```

Arguments

file	The HDF file path.
all	If TRUE, close all open HDF files.

Functions

- hql_use_file: Open (use) an HDF file.
- hql_close_file: Close an HDF file.

Examples

```
if(hql_is_loaded()){
    tf = tempfile(fileext = ".h5")
    hql_create_file(tf)

    hql_use_file(tf)
    hql_flush()

    hql_close_file(tf)
}
```

hql_flush*Flush HDF Files*

Description

Flush HDF file(s) to write buffered data to the disk.

Usage

```
hql_flush(global = TRUE)
```

Arguments

global	If TRUE, a global flush is performed and all open HDF files are flushed. If FALSE, a local flush is performed and only the HDF file currently in use is flushed.
--------	---

hql_is_loaded*HDFql Library Status*

Description

Check if the HDFql library loaded.

Usage

```
hql_is_loaded()
```

Value

Logical TRUE if DLLs are found, FALSE otherwise.

hql_list	<i>List HDF Groups, Datasets or Attributes</i>
----------	--

Description

List groups, datasets or attribute in an HDF file.

Usage

```
hql_list_groups(path, recursive = FALSE)

hql_list_datasets(path, recursive = FALSE)

hql_list_attributes(path)
```

Arguments

path	The location of the dataset, attribute, or group within the HDF file.
recursive	Recursively list child groups or datasets.

Value

A vector of paths.

Functions

- hql_list_groups: List groups.
- hql_list_datasets: List datasets.
- hql_list_attributes: List Attributes

hql_load	<i>Load HDFql DLLs</i>
----------	------------------------

Description

Load the HDFql library.

Usage

```
hql_load(path)

hql_unload()
```

Arguments

path	The path to the HDFql installation.
------	-------------------------------------

Functions

- hql_unload: Unload HDFqL Library.

hql_write	<i>Write HDF Dataset or Attribute</i>
-----------	---------------------------------------

Description

Write a dataset or attribute to an HDF file.

Usage

```
hql_write_dataset(
  dataset,
  path,
  include.attributes = TRUE,
  overwrite = FALSE,
  parallel = FALSE
)

hql_write_attribute(attribute, path, overwrite = FALSE, parallel = FALSE)

hql_write_all_attributes(attributes, path, overwrite = FALSE, parallel = FALSE)
```

Arguments

dataset	The dataset to write. The object must be coercible to an array.
path	The location within the HDF file to write the dataset or attribute(s).
include.attributes	If TRUE, write the dataset attributes.
overwrite	If TRUE, overwrite existing dataset or attribute.
parallel	Use parallel processing functionality.
attribute	The attribute to write.
attributes	A list of attributes to write.

Functions

- hql_write_dataset: Write a dataset to an HDF file.
- hql_write_attribute: Write an attribute to an HDF file.
- hql_write_all_attributes: Write multiple attributes to an HDF file.

Examples

```
if(hql_is_loaded()){
  tf = tempfile(fileext = ".h5")
  hql_create_file(tf)

  hql_use_file(tf)
  x = matrix(rnorm(100), nrow = 20)
  hql_write_dataset(x, "dataset0")
  hql_write_attribute("normal", "dataset0/dist")

  y = month.name
  attr(y, "abbreviation") = month.abb
  hql_write_dataset(y, "group1/dataset1")

  hql_close_file(tf)
}
```


Index

* **datasets**

hql, [2](#)

attach(), [2](#)

hdfqlr, [2](#)

hdfqlr-package (hdfqlr), [2](#)

hql, [2](#)

hql_close_file (hql_file), [4](#)

hql_create, [3](#)

hql_create_file (hql_create), [3](#)

hql_create_group (hql_create), [3](#)

hql_drop, [4](#)

hql_file, [4](#)

hql_flush, [5](#)

hql_is_loaded, [5](#)

hql_list, [6](#)

hql_list_attributes (hql_list), [6](#)

hql_list_datasets (hql_list), [6](#)

hql_list_groups (hql_list), [6](#)

hql_load, [6](#)

hql_unload (hql_load), [6](#)

hql_use_file (hql_file), [4](#)

hql_write, [7](#)

hql_write_all_attributes (hql_write), [7](#)

hql_write_attribute (hql_write), [7](#)

hql_write_dataset (hql_write), [7](#)

options(), [2](#)