Package 'parquetize'

March 13, 2023

Type Package

Title Convert Files to Parquet Format

Version 0.5.4

Description Collection of functions to get files in parquet format.
Parquet is a columnar storage file format <https://parquet.apache.org/>.
The files to convert can be of several formats
(``csv", ``RData", ``rds", ``RSQLite",
``json", ``ndjson", ``SAS", ``SPSS"...).

License Apache License (>= 2.0)

Encoding UTF-8

Depends R (>= 3.5.0)

URL https://ddotta.github.io/parquetize/,
https://github.com/ddotta/parquetize

BugReports https://github.com/ddotta/parquetize/issues

RoxygenNote 7.2.3

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Imports haven, arrow, curl, readr, jsonlite, DBI, RSQLite, cli, dplyr

VignetteBuilder knitr

NeedsCompilation no

Author Damien Dotta [aut, cre]

Maintainer Damien Dotta <damien.dotta@live.fr>

Repository CRAN

Date/Publication 2023-03-13 11:10:02 UTC

R topics documented:

`csv_to_parquet` *Convert a csv file to parquet format*

Description

This function allows to convert a csv file to parquet format.

Several conversion possibilities are offered :

- From a locally stored file. Argument 'path_to_csv' must then be used;
- From a URL. Argument 'url_to_csv' must then be used.

Two conversions possibilities are offered :

- Convert to a single parquet file. Argument 'path_to_parquet' must then be used;
- Convert to a partitioned parquet file. Additionnal arguments 'partition' and 'partitioning' must then be used;

Usage

```
csv_to_parquet(
  path_to_csv,
  url_to_csv,
  csv_as_a_zip = FALSE,
  filename_in_zip,
  path_to_parquet,
  columns = "all",
  compression = "snappy",
  compression_level = NULL,
  partition = "no",
  encoding = "UTF-8",
  ...
)
```

Arguments

path_to_csv	string that indicates the path to the csv file
url_to_csv	string that indicates the URL of the csv file
csv_as_a_zip	boolean that indicates if the csv is stored in a zip
filename_in_zip	name of the csv file in the zip (useful if several csv are included in the zip). Required if ‘csv_as_a_zip‘ is TRUE.
path_to_parquet	string that indicates the path to the directory where the parquet file will be stored
columns	character vector of columns to select from the input file (by default, all columns are selected).
compression	compression algorithm. Default "snappy".
compression_level	compression level. Meaning depends on compression algorithm.
partition	string ("yes" or "no" - by default) that indicates whether you want to create a partitioned parquet file. If "yes", ‘"partitioning"‘ argument must be filled in. In this case, a folder will be created for each modality of the variable filled in ‘"partitioning"‘.
encoding	string that indicates the character encoding for the input file.
...	additional format-specific arguments, see arrow::write_parquet() and arrow::write_dataset() for more informations.

Value

A parquet file, invisibly

Note

Be careful, if the zip size exceeds 4 GB, the function may truncate the data (because unzip() won't work reliably in this case - see here). In this case, it's advised to unzip your csv file by hand (for example with 7-Zip) then use the function with the argument ‘path_to_csv‘.

Examples

```
# Conversion from a local csv file to a single parquet file :

csv_to_parquet(
  path_to_csv = parquetize_example("region_2022.csv"),
  path_to_parquet = tempdir()
)

# Conversion from a local csv file to a single parquet file and select only
# fex columns :

csv_to_parquet(
```

```
  path_to_csv = parquetize_example("region_2022.csv"),
  path_to_parquet = tempdir(),
  columns = c("REG","LIBELLE")
)

# Conversion from a local csv file  to a partitioned parquet file  :

csv_to_parquet(
  path_to_csv = parquetize_example("region_2022.csv"),
  path_to_parquet = tempdir(),
  partition = "yes",
  partitioning =  c("REG")
)

# Conversion from a URL and a csv file with "gzip" compression :

csv_to_parquet(
  url_to_csv =
 "https://github.com/sidsriv/Introduction-to-Data-Science-in-python/raw/master/census.csv",
  path_to_parquet = tempdir(),
  compression = "gzip",
  compression_level = 5
)

# Conversion from a URL and a zipped file :

csv_to_parquet(
  url_to_csv = "https://www.nomisweb.co.uk/output/census/2021/census2021-ts007.zip",
  csv_as_a_zip = TRUE,
  filename_in_zip = "census2021-ts007-ctry.csv",
  path_to_parquet = tempdir()
)
```

`json_to_parquet` *Convert a json file to parquet format*

Description

This function allows to convert a json or ndjson file to parquet format.

Two conversions possibilities are offered :

- Convert to a single parquet file. Argument ‘path_to_parquet‘ must then be used;
- Convert to a partitioned parquet file. Additionnal arguments ‘partition‘ and ‘partitioning‘ must then be used;

Usage

```
json_to_parquet(
  path_to_json,
  path_to_parquet,
  format = "json",
  partition = "no",
  ...
)
```

Arguments

path_to_json	string that indicates the path to the csv file
path_to_parquet	string that indicates the path to the directory where the parquet file will be stored
format	string that indicates if the format is "json" (by default) or "ndjson"
partition	string ("yes" or "no" - by default) that indicates whether you want to create a partitioned parquet file. If "yes", ‘"partitioning"‘ argument must be filled in. In this case, a folder will be created for each modality of the variable filled in ‘"partitioning"‘.
...	additional format-specific arguments, see arrow::write_parquet() and arrow::write_dataset() for more informations.

Value

A parquet file, invisibly

Examples

```
# Conversion from a local json file to a single parquet file ::

json_to_parquet(
  path_to_json = system.file("extdata","iris.json",package = "parquetize"),
  path_to_parquet = tempdir()
)

# Conversion from a local ndjson file to a partitioned parquet file  ::

json_to_parquet(
  path_to_json = system.file("extdata","iris.ndjson",package = "parquetize"),
  path_to_parquet = tempdir(),
  format = "ndjson"
)
```

parquetize_example *Get path to parquetize example*

Description

parquetize comes bundled with a number of sample files in its 'inst/extdata' directory. This function make them easy to access

Usage

```
parquetize_example(file = NULL)
```

Arguments

file	Name of file. If 'NULL', the example files will be listed.

Value

A character string

Examples

```
parquetize_example()
parquetize_example("region_2022.csv")
```

rbind_parquet *Function to bind multiple parquet files by row*

Description

This function read all parquet files in 'folder' argument that starts with 'output_name', combine them using rbind and write the result to a new parquet file.

It can also delete the initial files if 'delete_initial_files' argument is TRUE.

Be careful, this function will not work if files with different structures are present in the folder given with the argument 'folder'.

Usage

```
rbind_parquet(folder, output_name, delete_initial_files = TRUE)
```

Arguments

folder	the folder where the initial files are stored
output_name	name of the output parquet file
delete_initial_files	Boolean. Should the function delete the initial files ? By default TRUE.

Value

Parquet files, invisibly

Examples

```
## Not run:
library(arrow)
if (file.exists('output')==FALSE) {
  dir.create("output")
}

file.create(fileext = "output/test_data1-4.parquet")
write_parquet(data.frame(
  x = c("a","b","c"),
  y = c(1L,2L,3L)
),
"output/test_data1-4.parquet")

file.create(fileext = "output/test_data4-6.parquet")
write_parquet(data.frame(
  x = c("d","e","f"),
  y = c(4L,5L,6L)
), "output/test_data4-6.parquet")

test_data <- rbind_parquet(folder = "output",
                           output_name = "test_data",
                           delete_initial_files = FALSE)

## End(Not run)
```

---

rds_to_parquet *Convert a rds file to parquet format*

Description

This function allows to convert a rds file to parquet format.

Two conversions possibilities are offered :

- Convert to a single parquet file. Argument ‘path_to_parquet‘ must then be used;
- Convert to a partitioned parquet file. Additionnal arguments ‘partition‘ and ‘partitioning‘ must then be used;

Usage

```
rds_to_parquet(path_to_rds, path_to_parquet, partition = "no", ...)
```

Arguments

path_to_rds	string that indicates the path to the rds file
path_to_parquet	string that indicates the path to the directory where the parquet file will be stored
partition	string ("yes" or "no" - by default) that indicates whether you want to create a partitioned parquet file. If "yes", ‘"partitioning"‘ argument must be filled in. In this case, a folder will be created for each modality of the variable filled in ‘"partitioning"‘.
...	additional format-specific arguments, see arrow::write_parquet() and arrow::write_dataset() for more informations.

Value

A parquet file, invisibly

Examples

```
# Conversion from a local rds file to a single parquet file ::

rds_to_parquet(
  path_to_rds = system.file("extdata","iris.rds",package = "parquetize"),
  path_to_parquet = tempdir()
)

# Conversion from a local rds file to a partitioned parquet file  ::

rds_to_parquet(
  path_to_rds = system.file("extdata","iris.rds",package = "parquetize"),
  path_to_parquet = tempdir(),
  partition = "yes",
  partitioning =  c("Species")
)
```

---

sqlite_to_parquet *Convert a sqlite file to parquet format*

Description

This function allows to convert a table from a sqlite file to parquet format.
The following extensions are supported : "db","sdb","sqlite","db3","s3db","sqlite3","sl3","db2","s2db","sqlite2","sl2".

Two conversions possibilities are offered :

- Convert to a single parquet file. Argument ‘path_to_parquet‘ must then be used;
- Convert to a partitioned parquet file. Additionnal arguments ‘partition‘ and ‘partitioning‘ must then be used;

Usage

```
sqlite_to_parquet(
  path_to_sqlite,
  table_in_sqlite,
  path_to_parquet,
  partition = "no",
  ...
)
```

Arguments

`path_to_sqlite`	string that indicates the path to the sqlite file
`table_in_sqlite`	string that indicates the name of the table to convert in the sqlite file
`path_to_parquet`	string that indicates the path to the directory where the parquet file will be stored
`partition`	string ("yes" or "no" - by default) that indicates whether you want to create a partitioned parquet file. If "yes", ‘"partitioning"‘ argument must be filled in. In this case, a folder will be created for each modality of the variable filled in ‘"partitioning"‘.
`...`	additional format-specific arguments, see arrow::write_parquet() and arrow::write_dataset() for more informations.

Value

A parquet file, invisibly

Examples

```
# Conversion from a local sqlite file to a single parquet file :

sqlite_to_parquet(
  path_to_sqlite = system.file("extdata","iris.sqlite",package = "parquetize"),
  table_in_sqlite = "iris",
  path_to_parquet = tempdir()
)

# Conversion from a local sqlite file to a partitioned parquet file  :

sqlite_to_parquet(
  path_to_sqlite = system.file("extdata","iris.sqlite",package = "parquetize"),
  table_in_sqlite = "iris",
  path_to_parquet = tempdir(),
```

```
  partition = "yes",
  partitioning =  c("Species")
)
```

`table_to_parquet` *Convert an input file to parquet format*

Description

This function allows to convert an input file to parquet format.

It handles SAS, SPSS and Stata files in a same function. There is only one function to use for these 3 cases. For these 3 cases, the function guesses the data format using the extension of the input file (in the ‘path_to_table‘ argument).

Two conversions possibilities are offered :

- Convert to a single parquet file. Argument ‘path_to_parquet‘ must then be used;
- Convert to a partitioned parquet file. Additionnal arguments ‘partition‘ and ‘partitioning‘ must then be used;

To avoid overcharging R’s RAM, the conversion can be done by chunk. Argument ‘by_chunk‘ must then be used. This is very useful for huge tables and for computers with little RAM because the conversion is then done with less memory consumption. For more information, see [here](https://ddotta.github.io/parquetize/articles/aa-conversions.html).

Usage

```
table_to_parquet(
  path_to_table,
  path_to_parquet,
  columns = "all",
  by_chunk = FALSE,
  chunk_size,
  skip = 0,
  partition = "no",
  encoding = NULL,
  ...
)
```

Arguments

`path_to_table`	string that indicates the path to the input file (don’t forget the extension).
`path_to_parquet`	string that indicates the path to the directory where the parquet files will be stored.

columns	character vector of columns to select from the input file (by default, all columns are selected).
by_chunk	Boolean. By default FALSE. If TRUE then it means that the conversion will be done by chunk.
chunk_size	this argument must be filled in if ‘by_chunk‘ is TRUE. Number of lines that defines the size of the chunk.
skip	By default 0. This argument must be filled in if ‘by_chunk‘ is TRUE. Number of lines to ignore when converting.
partition	string ("yes" or "no" - by default) that indicates whether you want to create a partitioned parquet file. If "yes", ‘"partitioning"‘ argument must be filled in. In this case, a folder will be created for each modality of the variable filled in ‘"partitioning"‘. Be careful, if ‘by_chunk‘ argument is not NULL then a single parquet file will be created.
encoding	string that indicates the character encoding for the input file.
...	additional format-specific arguments, see arrow::write_parquet() and arrow::write_dataset() for more informations.

Value

Parquet files, invisibly

Examples

```
# Conversion from a SAS file to a single parquet file :

table_to_parquet(
  path_to_table = system.file("examples","iris.sas7bdat", package = "haven"),
  path_to_parquet = tempdir()
)

# Conversion from a SPSS file to a single parquet file :

table_to_parquet(
  path_to_table = system.file("examples","iris.sav", package = "haven"),
  path_to_parquet = tempdir(),
)
# Conversion from a Stata file to a single parquet file without progress bar :

table_to_parquet(
  path_to_table = system.file("examples","iris.dta", package = "haven"),
  path_to_parquet = tempdir()
)

# Reading SAS file by chunk and with encoding and conversion
# from a SAS file to a single parquet file :

table_to_parquet(
  path_to_table = system.file("examples","iris.sas7bdat", package = "haven"),
  path_to_parquet = tempdir(),
```

```
  by_chunk = TRUE,
  chunk_size = 50,
  encoding = "utf-8"
)

# Conversion from a SAS file to a single parquet file and select only
#few columns  :

table_to_parquet(
  path_to_table = system.file("examples","iris.sas7bdat", package = "haven"),
  path_to_parquet = tempdir(),
  columns = c("Species","Petal_Length")
)

# Conversion from a SAS file to a partitioned parquet file  :

table_to_parquet(
  path_to_table = system.file("examples","iris.sas7bdat", package = "haven"),
  path_to_parquet = tempdir(),
  partition = "yes",
  partitioning =  c("Species") # vector use as partition key
)
```

Index