Package 'fastreg'

June 18, 2026

Title Fast Conversion and Querying of Danish Registers with 'Parquet'

Version 0.14.1

Description Converts large Danish register files ('sas7bdat') into 'Parquet' format with year-based 'Hive' partitioning and chunked reading for larger-than-memory files. Supports parallel conversion with a 'targets' pipeline and reading those registers into 'DuckDB' tables for faster querying and analyses.

License MIT + file LICENSE

URL https://dp-next.github.io/fastreg/
https://github.com/dp-next/fastreg

BugReports https://github.com/dp-next/fastreg/issues

Depends R (>= 4.1.0)

Imports arrow, checkmate, cli, dplyr, fs, glue, haven, osdc, purrr, rlang, stringr, tibble, uuid

Suggests crew, tidyr, dbplyr, devtools, duckdb, knitr, qs2, quarto, targets, testthat (>= 3.0.0), tidyselect, withr, tarchetypes

VignetteBuilder quarto

Encoding UTF-8

Language en-US

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Signe Kirk Brødbæk [aut, cre] (ORCID: <https://orcid.org/0009-0000-2208-7088>), Luke Johnston [aut] (ORCID: <https://orcid.org/0000-0003-4169-2616>), Steno Diabetes Center Aarhus [cph], Aarhus University [cph]

Maintainer Signe Kirk Brødbæk <signekb@clin.au.dk>

Repository CRAN

Date/Publication 2026-06-18 11:30:14 UTC

Contents

`convert` *Convert a single register SAS file to Parquet*

Description

To be able to handle larger-than-memory files, the SAS file is converted in chunks. It does not check for existing files in the output directory. Existing data will not be overwritten, but might be duplicated if it already exists in the directory, since files are saved with UUIDs in their names.

Usage

```
convert(path, output_dir, chunk_size = 10000000L)
```

Arguments

`path`	Path to a single SAS file.
`output_dir`	Directory to save the Parquet output to. Must not include the register name as this will be extracted from `path` to create the register folder.
`chunk_size`	Number of rows to read and convert at a time.

Details

On Windows, `haven::read_sas()` silently re-reads the first chunk when `skip` exceeds 2,147,483,647 (the 32-bit integer limit). `convert()` detects this and stops the conversion with a warning, so the remainder of the file is not converted.

Value

A tibble with a conversion log about each written chunk.

See Also

Getting started and the When SAS files become too big section for handling SAS files with more than 2,147,483,647 rows.

Examples

```
sas_file <- fs::path_package("fastreg", "extdata", "test.sas7bdat")
convert(
  path = sas_file,
  output_dir = fs::path_temp("path/to/output/file")
)
```

---

list_parquet *List Parquet datasets or files in a project*

Description

Only lists Parquet files that end in `part-*.parquet`. For datasets, it will only look for Parquet files with a `year=YYYY` in its path. This function will search the whole system for the project ID, so it might be slow sometimes.

Usage

```
list_parquet_datasets()

list_parquet_files()
```

Value

The path(s) to the Parquet datasets (as directories) or files.

Functions

- `list_parquet_datasets()`: List all Parquet (Hive partitioned by year) datasets.
- `list_parquet_files()`: List all Parquet files within a project.

---

list_sas_files *List SAS files in a directory*

Description

Lists all SAS register files (with the extension `.sas7bdat` case-insensitively) in the specified directory and its subdirectories.

Usage

```
list_sas_files(path)
```

Arguments

`path`	Directory to search.

Value

The path(s) to the found SAS file(s).

Examples

```
list_sas_files(fs::path_package("fastreg", "extdata"))
```

`print_log_row_count` *Log chunk information as a table*

Description

Turns the log information returned by `convert()` into a pretty table, showing relative input/output paths and row counts.

Usage

```
print_log_row_count(log)
```

Arguments

`log`	A tibble returned by `convert()`, with columns `input_path`, `output_path`, and `row_count`.

Value

`log` invisibly.

Examples

```
sas_file <- fs::path_package("fastreg", "extdata", "test.sas7bdat")
conversion_log <- convert(sas_file, output_dir = fs::path_temp("output"))
print_log_row_count(conversion_log)
```

`print_log_schema` *Print log schema comparison*

Description

Prints the log schema information in a section that compares the schemas within one register. Finds the most common schema and if there's differences between schemas, it prints these differences.

Usage

```
print_log_schema(register_log)
```

Arguments

register_log	A tibble returned by convert(), filtered to only contain rows from a single register.

Value

register_log invisibly.

Examples

```
sas_file <- fs::path_package("fastreg", "extdata", "test.sas7bdat")
log <- convert(sas_file, output_dir = fs::path_temp("output"))
print_log_schema(log)
```

---

read_parquet *Read a single Parquet file or a partitioned dataset as DuckDB table*

Description

This is useful when the read_register() incorrectly guesses or can't find the register.

Usage

```
read_parquet_dataset(path)

read_parquet_file(path)
```

Arguments

path	Path to a directory with the Parquet files within or a path to a Parquet file.

Value

A DuckDB table.

Functions

- read_parquet_dataset(): Reads a Parquet partitioned directory.
- read_parquet_file(): Reads a single Parquet file.

read_register *Read a Parquet register*

Description

This function uses the options `fastreg.project_rawdata_dir` and `fastreg.project_workdata_dir` when set in `options()` or will try to guess the path by using the project ID and the base directories `E:/<project-id>/rawdata/` and `E:/<project-id>/workdata/`. It only reads Parquet datasets (those that are partitioned with the pattern `year=`). If this function doesn't work, use `read_parquet_dataset()` or `read_parquet_file()` instead.

Usage

```
read_register(name)
```

Arguments

`name`	Name of the Parquet dataset (i.e, the register name). See a list of available datasets with `list_parquet_datasets()`.

Value

A DuckDB table.

simulate_registers_with_paths *Simulate example registers along with output paths for SAS files*

Description

A helper function that simulates data using `osdc::simulate_registers()`. It's used in vignettes and tests. It simulates data for one or more registers and years.

Usage

```
simulate_registers_with_paths(
  registers,
  years = "",
  n = 1000,
  output_dir = fs::path_temp("E/rawdata/701010/")
)
```

Arguments

registers	Name of one or more registers. Must be a register that osdc::simulate_registers() can simulate. See osdc::registers() for a list of available registers.
years	One or more years to save the simulated data under. The year is used as a suffix in the file name. For example for register "bef" and year "1999", the file will be named bef1999.sas7bdat. Can also take no year.
n	Number of rows of data to simulate per year.
output_dir	The root directory appended to the created SAS paths. By default, the output_dir is a temp path that mimics the paths on DST, E/rawdata/701010. The default should technically be E: on Windows, but the default temporary directory on Windows for R doesn't allow using :, so we use E instead.

Value

A nested tibble with a column data containing the simulated data and a column output_path containing the path where the SAS file should be saved to. Pipe to purrr::pwalk(write_to_sas) or purrr::pmap(write_to_sas) to write each simulated dataset to a SAS file.

Examples

```
sim_regs <- simulate_registers_with_paths(
  registers = c("bef", "lmdb"),
  years = c("1999", "2000"),
  n = 10,
)
sim_regs

sim_regs |>
  purrr::pwalk(write_to_sas)
```

---

use_template *Use a targets pipeline for converting SAS registers to Parquet*

Description

Copies a _targets.R template and a conversion log Quarto Markdown file to the given directory.

Usage

```
use_template(path = ".", open = rlang::is_interactive())
```

Arguments

path	Path to the directory where the targets pipeline and conversion log will be created. Defaults to the current directory.
open	Whether to open the file for editing.

Value

The path to the created `_targets.R` file, invisibly.

Examples

```
use_template(path = fs::path_temp(""))
```

`write_to_sas` *Write simulated data to a SAS file*

Description

A helper function that writes a data frame to a SAS file. It's used mainly in fastreg's vignettes and tests. Pipe the output of `simulate_registers_with_paths()` with `purrr::pwalk()` followed by this function to write each simulated dataset to a SAS file.

Usage

```
write_to_sas(data, output_path)
```

Arguments

`data`	A tibble containing the simulated data.
`output_path`	A string of the path to where the SAS file should be saved.

Value

Invisibly gives the path to the saved SAS file.

Index