

Package ‘datasusr’

May 8, 2026

Type Package

Title Fast Access to Brazilian Public Health Data from 'DATASUS'

Version 0.1.0

Description Provides fast, in-memory reading of 'DATASUS' 'DBC' files using native 'C' code, along with a catalog of public health data sources, 'FTP' file discovery, caching downloads, and a high-level `datasusr_fetch()` function that lists, downloads, and reads files in a single call. Bundles the 'blast' decompressor from 'zlib' contrib/blast to decode 'PKWare DCL' compressed 'DBC' files and parses 'DBF' records directly for efficient import into tibbles. See the 'DATASUS' file transfer site <https://datasus.saude.gov.br> and Adler (2003) <https://github.com/madler/zlib/tree/master/contrib/blast> for details on the underlying data and compression format.

License MIT + file LICENSE

URL <https://strategicprojects.github.io/datasusr/>,
<https://github.com/StrategicProjects/datasusr>

BugReports <https://github.com/StrategicProjects/datasusr/issues>

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Imports cli, curl (>= 5.0.0), dplyr (>= 1.1.0), purrr, rlang (>= 1.0.0), stringr, tibble, tidyr

Suggests knitr, pkgdown, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

RoxygenNote 7.3.3

NeedsCompilation yes

SystemRequirements C99

Author Andre Leite [aut, cre],
 Marcos Wasilew [aut],
 Hugo Vasconcelos [aut],
 Carlos Amorin [aut],
 Diogo Bezerra [aut],
 Mark Adler [ctb, cph] (Author of bundled blast.c and blast.h from zlib
 contrib/blast)

Maintainer Andre Leite <leite@castlab.org>

Repository CRAN

Date/Publication 2026-05-04 11:50:02 UTC

Contents

datasus_build_path	2
datasus_cache_clear	3
datasus_cache_dir	4
datasus_cache_info	4
datasus_cache_list	5
datasus_cache_prune	5
datasus_docs_url	6
datasus_download	7
datasus_fetch	8
datasus_file_types	9
datasus_ftp_ls	10
datasus_get_territory	11
datasus_list_files	12
datasus_modalities	13
datasus_sources	13
datasus_ufs	14
format_bytes	14
read_datasus_dbc	15

Index 17

datasus_build_path *Build DATASUS FTP paths from filters*

Description

Builds candidate FTP directories for a given source and file type. For SIH/SIA, the function selects historical vs. current trees by year/month. For SIM and SINASC, preliminary trees are included when requested.

Usage

```
datusus_build_path(
  source,
  file_type,
  year = NULL,
  month = NULL,
  include_prelim = TRUE
)
```

Arguments

source	Source code (e.g. "SIHSUS").
file_type	File type code (e.g. "RD").
year	Integer vector of years.
month	Integer vector of months.
include_prelim	Logical. Include preliminary trees when available (default TRUE).

Value

A tibble with columns source, file_type, period, and path.

Examples

```
datusus_build_path(source = "SIHSUS", file_type = "RD", year = 2024, month = 1)
```

datusus_cache_clear *Clear cached DATASUS files*

Description

Removes files from the cache directory. By default all cached files are removed; pass a character vector to files to remove specific paths.

Usage

```
datusus_cache_clear(cache_dir = NULL, files = NULL, verbose = TRUE)
```

Arguments

cache_dir	Optional cache directory.
files	Optional character vector of file paths to remove. When NULL, all cached files are removed.
verbose	Logical. Emit progress messages (default TRUE).

Value

A tibble with columns path and removed.

datasus_cache_dir	<i>Resolve the datasusr cache directory</i>
-------------------	---

Description

Returns the active cache directory path. Checks, in order: the `cache_dir` argument, the `DATASUSR_CACHE_DIR` environment variable, the `datasusr.cache_dir` option, and finally a session-scoped subdirectory of `tempdir()`. To opt in to a persistent cache across sessions, set the `DATASUSR_CACHE_DIR` environment variable, the `datasusr.cache_dir` option, or pass `cache_dir` explicitly (for example, `tools::R_user_dir("datasusr", "cache")`).

Usage

```
datasus_cache_dir(cache_dir = NULL)
```

Arguments

`cache_dir` Optional cache directory supplied by the caller.

Value

A single path string.

Examples

```
datasus_cache_dir()
```

datasus_cache_info	<i>Summarise the datasusr cache</i>
--------------------	-------------------------------------

Description

Returns a one-row tibble summarising the current cache: directory path, file count, total size, and modification time range.

Usage

```
datasus_cache_info(cache_dir = NULL, verbose = TRUE)
```

Arguments

`cache_dir` Optional cache directory.
`verbose` Logical. Print a summary to the console (default TRUE).

Value

A tibble with one row.

Examples

```
datasus_cache_info()
```

```
datasus_cache_list    List cached DATASUS files
```

Description

Returns a tibble describing every file currently stored in the cache directory.

Usage

```
datasus_cache_list(cache_dir = NULL)
```

Arguments

`cache_dir` Optional cache directory.

Value

A tibble with columns `path`, `file_name`, `size_bytes`, `modified_time`, and `accessed_time`.

Examples

```
datasus_cache_list()
```

```
datasus_cache_prune  Prune the datasusr cache
```

Description

Selectively removes cached files based on age and/or total size. Older and least-recently-accessed files are removed first.

Usage

```
datasus_cache_prune(  
  cache_dir = NULL,  
  max_size_bytes = NULL,  
  older_than_days = NULL,  
  verbose = TRUE  
)
```

Arguments

cache_dir	Optional cache directory.
max_size_bytes	Maximum total cache size in bytes. When exceeded, the least-recently-accessed files are removed until the cache fits.
older_than_days	Age threshold in days. Files with a modification time older than this are removed.
verbose	Logical. Emit progress messages (default TRUE).

Value

A tibble with columns path, removed, and reason.

datasus_docs_url	<i>Get FTP URLs for DATASUS documentation</i>
------------------	---

Description

Returns the FTP URLs where documentation files (layouts, data dictionaries) for each source system can be found. Use `datasus_ftp_ls()` to list the actual files at each URL.

Usage

```
datasus_docs_url(source = NULL)
```

Arguments

source	Optional source code to filter (e.g. "SIHSUS"). When NULL, all known documentation paths are returned.
--------	--

Value

A tibble with columns source and docs_url.

Examples

```
datasus_docs_url()
datasus_docs_url("CNES")

tryCatch({
  # List documentation files for CNES (network required).
  docs <- datasus_docs_url("CNES")
  datasus_ftp_ls(docs$docs_url[[1]], verbose = FALSE)
}, error = function(e) message("FTP unavailable: ", conditionMessage(e)))
```

datasus_download	<i>Download DATASUS files</i>
------------------	-------------------------------

Description

Downloads one or many DATASUS files. When `use_cache = TRUE`, files that already exist in the cache directory are reused instead of re-downloaded.

Usage

```
datasus_download(  
  files = NULL,  
  ...,  
  dest_dir = NULL,  
  overwrite = FALSE,  
  timeout = 240,  
  use_cache = TRUE,  
  cache_dir = NULL,  
  refresh = FALSE,  
  verbose = TRUE  
)
```

Arguments

<code>files</code>	A tibble returned by <code>datasus_list_files()</code> . When <code>NULL</code> , additional filters are forwarded to <code>datasus_list_files()</code> .
<code>...</code>	Filters passed to <code>datasus_list_files()</code> when <code>files</code> is <code>NULL</code> .
<code>dest_dir</code>	Optional destination directory. When <code>NULL</code> and <code>use_cache = TRUE</code> , the package cache directory is used.
<code>overwrite</code>	Logical. Overwrite existing files (default <code>FALSE</code>).
<code>timeout</code>	Timeout in seconds for each download (default 240).
<code>use_cache</code>	Logical. Store and reuse downloads in the cache directory (default <code>TRUE</code>).
<code>cache_dir</code>	Optional cache directory.
<code>refresh</code>	Logical. Force re-download even when a cached file exists (default <code>FALSE</code>).
<code>verbose</code>	Logical. Emit progress messages (default <code>TRUE</code>).

Value

A tibble with a `local_file` column containing the paths to the downloaded files, plus a downloaded flag.

Examples

```
tryCatch({
  files <- datasus_list_files(
    source = "SIHSUS", file_type = "RD",
    year = 2024, month = 1, uf = "AC",
    verbose = FALSE
  )
  downloads <- datasus_download(
    files,
    cache_dir = tempdir(),
    verbose = FALSE
  )
}, error = function(e) message("FTP unavailable: ", conditionMessage(e)))
```

 datasus_fetch

Fetch DATASUS data in one step

Description

A convenience wrapper that lists, downloads, and reads DATASUS files in a single call. Particularly useful for interactive / exploratory work.

Usage

```
datasus_fetch(
  source,
  file_type,
  year = NULL,
  month = NULL,
  uf = NULL,
  ...,
  bind = TRUE,
  include_prelim = TRUE,
  timeout = 240,
  use_cache = TRUE,
  cache_dir = NULL,
  verbose = TRUE
)
```

Arguments

source	Character vector of source codes.
file_type	Character vector of file type codes.
year	Integer vector of years.
month	Integer vector of months (required for monthly sources).
uf	Character vector of UF codes (required for UF-scoped sources).

...	Additional arguments forwarded to <code>read_datasus_dbc()</code> (e.g. <code>select</code> , <code>col_types</code> , <code>parse_dates</code>).
<code>bind</code>	Logical. When TRUE (the default), all files are row-bound into a single tibble. When FALSE, a list of tibbles is returned.
<code>include_prelim</code>	Logical. Include preliminary data trees (default TRUE).
<code>timeout</code>	Timeout in seconds for FTP and download operations (default 240).
<code>use_cache</code>	Logical. Reuse cached downloads (default TRUE).
<code>cache_dir</code>	Optional cache directory.
<code>verbose</code>	Logical. Emit progress messages (default TRUE).

Value

A tibble (when `bind = TRUE`) or a named list of tibbles.

Examples

```
tryCatch({
  # Fetch a small SIHSUS slice into tempdir() (network required).
  df <- datasus_fetch(
    source      = "SIHSUS",
    file_type   = "RD",
    year        = 2024,
    month       = 1,
    uf          = "AC",
    select      = c("uf_zi", "ano_cmpt", "munic_res", "val_tot"),
    cache_dir   = tempdir(),
    verbose     = FALSE
  )
}, error = function(e) message("FTP unavailable: ", conditionMessage(e)))
```

`datasus_file_types` *List DATASUS file types*

Description

Returns the internal catalog of file types. Optionally filtered by source and/or file type codes.

Usage

```
datasus_file_types(source = NULL, file_type = NULL)
```

Arguments

<code>source</code>	Optional character vector of source codes (e.g. "SIHSUS").
<code>file_type</code>	Optional character vector of file type codes (e.g. "RD").

Value

A tibble.

Examples

```
datasus_file_types()
datasus_file_types(source = "SIHSUS")
datasus_file_types(source = "CNES", file_type = "ST")
```

datasus_ftp_ls	<i>List files and directories from the DATASUS FTP</i>
----------------	--

Description

Fetches a raw directory listing from a DATASUS FTP path.

Usage

```
datasus_ftp_ls(path, timeout = 120, ftp_use_epsv = FALSE, verbose = TRUE)
```

Arguments

path	FTP path or full URL. Relative paths are prefixed with the DATASUS public FTP root.
timeout	Timeout in seconds (default 120).
ftp_use_epsv	Logical. Passed to curl (default FALSE).
verbose	Logical. Emit progress messages (default TRUE).

Value

A tibble with columns ftp_url and entry.

Examples

```
tryCatch(
  datasus_ftp_ls("SIHSUS/200801_/Dados/", verbose = FALSE),
  error = function(e) message("FTP unavailable: ", conditionMessage(e))
)
```

datasus_get_territory *Download DATASUS territorial tables*

Description

Downloads and reads tables from the DATASUS territorial data section. The DATASUS FTP publishes a ZIP archive per year containing reference tables for municipalities, health regions, and other geographic divisions used by SUS, in CSV, DBF, and TXT formats.

Usage

```
datasus_get_territory(  
  table = "tb_municip",  
  year = NULL,  
  format = "csv",  
  cache_dir = NULL,  
  verbose = TRUE  
)
```

Arguments

table	Name of the table to read. Common values: "tb_municip" (municipalities), "tb_uf" (states), "tb_regsaud" (health regions), "tb_macsaud" (macro health regions), "tb_regiao" (geographic regions), "tb_micibge" (IBGE micro-regions), "tb_regmetr" (metropolitan regions), "tb_dsei" (indigenous health districts). Relationship tables start with "rl_" (e.g. "rl_municip_regsaud").
year	Year of the territorial table. Defaults to the current year. Use <code>datasus_ftp_ls("ftp://ftp.datasus.gov.br/territorio/anos/anos.zip")</code> to see available years.
format	File format to extract from the ZIP: "csv" (default) or "dbf".
cache_dir	Optional cache directory.
verbose	Logical. Emit progress messages (default TRUE).

Value

A tibble with column names in snake_case.

Examples

```
tryCatch({  
  # Download territorial tables into tempdir() (network required).  
  municipios <- datasus_get_territory(  
    "tb_municip",  
    cache_dir = tempdir(),  
    verbose = FALSE  
  )  
  ufs <- datasus_get_territory(  
    "tb_uf",
```

```

    cache_dir = tempdir(),
    verbose   = FALSE
  )
}, error = function(e) message("FTP unavailable: ", conditionMessage(e)))

```

datasus_list_files *List available DATASUS files*

Description

Builds candidate file names from the internal catalog and, optionally, validates them against the DATASUS FTP.

Usage

```

datasus_list_files(
  source,
  file_type,
  year = NULL,
  month = NULL,
  uf = NULL,
  include_prelim = TRUE,
  check_exists = TRUE,
  timeout = 120,
  ftp_use_epsv = FALSE,
  verbose = TRUE
)

```

Arguments

source	Character vector of source codes.
file_type	Character vector of file type codes.
year	Integer vector of years.
month	Integer vector of months (required for monthly sources).
uf	Character vector of UF codes (required for UF-scoped sources).
include_prelim	Logical. Include preliminary data trees (default TRUE).
check_exists	Logical. Query the FTP and keep only files that exist (default TRUE). Setting to FALSE skips FTP access and returns all candidate files.
timeout	Timeout in seconds for FTP requests (default 120).
ftp_use_epsv	Logical. Passed to curl (default FALSE).
verbose	Logical. Emit progress messages (default TRUE).

Value

A tibble with one row per file, including its FTP URL and metadata.

Examples

```
tryCatch(  
  datasus_list_files(  
    source = "SIHSUS",  
    file_type = "RD",  
    year = 2024,  
    month = 1,  
    uf = c("PE", "PB"),  
    verbose = FALSE  
  ),  
  error = function(e) message("FTP unavailable: ", conditionMessage(e))  
)
```

datasus_modalities *List DATASUS modalities*

Description

Returns the available DATASUS data modalities (data, documentation, programs, etc.).

Usage

```
datasus_modalities()
```

Value

A tibble.

Examples

```
datasus_modalities()
```

datasus_sources *List DATASUS data sources*

Description

Returns the internal catalog of DATASUS sources available in datasusr.

Usage

```
datasus_sources()
```

Value

A tibble with one row per source, including its code, description, default scope, and flags for monthly and UF support.

Examples

```
datasus_sources()
```

datasus_ufs	<i>List UF codes used by DATASUS downloads</i>
-------------	--

Description

Returns a character vector of the 27 Brazilian state abbreviations accepted by DATASUS file naming conventions.

Usage

```
datasus_ufs()
```

Value

A character vector of length 27.

Examples

```
datasus_ufs()
```

format_bytes	<i>Format byte sizes for display</i>
--------------	--------------------------------------

Description

Converts raw byte counts into human-readable strings (e.g. "1.23 MB").

Usage

```
format_bytes(x)
```

Arguments

x Numeric vector of byte sizes.

Value

A character vector with formatted sizes.

Examples

```
format_bytes(c(1024, 1048576, NA))
```

read_datasus_dbc	<i>Read DATASUS DBC files</i>
------------------	-------------------------------

Description

Reads a DATASUS .dbc file directly into R using native C code for PKWare DCL decompression and DBF parsing. The function always returns a tibble and is designed to work well with the tidyverse.

Usage

```
read_datasus_dbc(
  file,
  select = NULL,
  n_max = Inf,
  trim_ws = TRUE,
  encoding = "latin1",
  guess_types = TRUE,
  col_types = NULL,
  parse_dates = FALSE,
  clean_names = TRUE,
  verbose = TRUE
)
```

Arguments

file	Path to a .dbc or .dbf file. Both compressed (DBC) and uncompressed (DBF) formats are accepted.
select	Optional character vector of column names to keep. When NULL (the default), all columns are returned. Names are matched case-insensitively, so both "UF_ZI" and "uf_zi" work.
n_max	Maximum number of rows to read. Defaults to Inf (all rows).
trim_ws	Logical. Trim leading/trailing whitespace from character fields (default TRUE).
encoding	Encoding of the DBF character fields. Typically "latin1" (the default for DATASUS files) or "UTF-8".
guess_types	Logical. Inspect numeric fields to distinguish integer-like columns from double columns (default TRUE). Disable for faster reads when precise types are not needed.
col_types	Optional named character vector of explicit column types. Supported values: "character", "integer", "double", "logical", "date". Names are matched case-insensitively against the original DBF field names.
parse_dates	Logical. If TRUE, DBF date fields (D type) are returned as Date. Character fields are only parsed as dates when explicitly listed in col_types (default FALSE).

clean_names	Logical. If TRUE (the default), column names are converted to snake_case (lowercase with underscores). The original DATASUS names are uppercase (e.g. UF_ZI becomes uf_zi).
verbose	Logical. Emit progress messages (default TRUE).

Value

A tibble.

Examples

```
# The example downloads a small DATASUS file into tempdir() and reads it.
# Skipped automatically if the FTP is unreachable.
tmp <- file.path(tempdir(), "RDAC2401.dbc")
url <- paste0(
  "ftp://ftp.datasus.gov.br/dissemin/publicos/SIHSUS/200801_/Dados/",
  "RDAC2401.dbc"
)
ok <- tryCatch(
  {
    utils::download.file(url, tmp, mode = "wb", quiet = TRUE)
    TRUE
  },
  error = function(e) FALSE,
  warning = function(w) FALSE
)

if (ok) {
  # Basic read (column names in snake_case by default)
  x <- read_datasus_dbc(tmp, verbose = FALSE)

  # Select works with either case
  x <- read_datasus_dbc(
    tmp,
    select = c("uf_zi", "ano_cmpt", "val_tot"),
    verbose = FALSE
  )

  # Keep original uppercase names
  x <- read_datasus_dbc(tmp, clean_names = FALSE, verbose = FALSE)

  unlink(tmp)
}
```


Index

`datasus_build_path`, 2
`datasus_cache_clear`, 3
`datasus_cache_dir`, 4
`datasus_cache_info`, 4
`datasus_cache_list`, 5
`datasus_cache_prune`, 5
`datasus_docs_url`, 6
`datasus_download`, 7
`datasus_fetch`, 8
`datasus_file_types`, 9
`datasus_ftp_ls`, 10
`datasus_ftp_ls()`, 6
`datasus_get_territory`, 11
`datasus_list_files`, 12
`datasus_list_files()`, 7
`datasus_modalities`, 13
`datasus_sources`, 13
`datasus_ufs`, 14

`format_bytes`, 14

`read_datasus_dbc`, 15
`read_datasus_dbc()`, 9

`tempdir()`, 4