

Package ‘readabs’

May 18, 2025

Type Package

Title Download and Tidy Time Series Data from the Australian Bureau of Statistics

Version 0.4.19

Maintainer Matt Cowgill <mattcowgill@gmail.com>

Description Downloads, imports, and tidies time series data from the Australian Bureau of Statistics <<https://www.abs.gov.au/>>.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 3.5)

Imports readxl (>= 1.2.0), dplyr (>= 0.8.0), hutils (>= 1.5.0), fst, purrr (>= 1.0.0), tidyr (>= 1.0.0), stringi, tools, glue, httr, rvest, xml2, rlang, labelled

URL <https://github.com/mattcowgill/readabs>

BugReports <https://github.com/mattcowgill/readabs/issues>

RoxygenNote 7.3.2

VignetteBuilder knitr

Suggests knitr, rmarkdown, markdown, testthat (>= 2.1.0), ggplot2

NeedsCompilation no

Author Matt Cowgill [aut, cre] (ORCID: <<https://orcid.org/0000-0003-0422-3300>>),
Zoe Meers [aut],
Jaron Lee [aut],
David Diviny [aut],
Hugh Parsonage [ctb],
Kinto Behr [ctb],
Angus Moore [ctb],
Francis Markham [ctb] (ORCID: <<https://orcid.org/0000-0002-4266-2569>>)

Repository CRAN

Date/Publication 2025-05-18 07:00:02 UTC

Contents

abs_api	2
check_latest_date	4
download_abs_data_cube	6
extract_abs_sheets	7
get_available_ifs_cubes	8
read_abs	8
read_abs_data	11
read_abs_local	11
read_abs_metadata	12
read_abs_url	13
read_awe	14
read_cpi	15
read_erp	16
read_job_mobility	17
read_ifs_datacube	18
read_ifs_grossflows	19
read_payrolls	20
scrape_abs_catalogues	21
search_catalogues	21
search_files	22
separate_series	23
show_available_catalogues	24
show_available_files	24
tidy_abs	25
tidy_abs_list	26
Index	27

abs_api

ABS.Stat API functions

Description

[Experimental]

These experimental functions provide a minimal interface to the ABS.Stat API.

More information on the ABS.Stat API can be found on the [ABS website](#)

Note that an ABS.Stat 'dataflow' is like a table. A 'datastructure' contains metadata that describes the variables in the dataflow. To load data from the ABS.Stat API, you need to either:

- Using `read_api_dataflows()` you can get information on the available dataflows
- Using `read_api_datastructure()` you can get metadata relating to a specific dataflow, including the variables available in each dataflow
- Using `read_api()` you can get the data belonging to a given dataflow.
- Using `read_api_url()` you can get the data for a given query url generated using the [online data viewer](#).

Usage

```

read_api_dataflows()

read_api(
  id,
  datakey = NULL,
  start_period = NULL,
  end_period = NULL,
  version = NULL
)

read_api_url(url)

read_api_datastructure(id)

```

Arguments

id	A dataflow id. Use <code>read_api_dataflows()</code> to obtain a dataframe listing available dataflows.
datakey	A named list matching filter variables to codes. All variables with a position in the datastructure are filterable. Use <code>read_api_datastructure()</code> to obtain information about the variables in a dataflow and the values of that variable.
start_period	The start period (used to filter by time). This is inclusive. The supported formats are: <ul style="list-style-type: none"> • "YYYY" for annual data (e.g. 2019) • "YYYY-S[1-2]" for semi-annual data (e.g. 2019-S1) • "YYYY-Q[1-4]" for quarterly data (e.g. 2019-Q1) • "YYYY-MM[01-12]" for monthly data (e.g. 2019-01) • "YYYY-W[01-53]" for weekly data (e.g. 2019-W01) • "YYYY-MM-DD" for daily and business data (e.g. 2019-01-01)
end_period	The end period (used to filter on time). This is inclusive. The supported formats are the same as for <code>start_period</code>
version	A version number, if unspecified the latest version of the dataset is used. Use <code>read_api_dataflows()</code> to see available dataflow versions.
url	A complete query url

Details

Note that the API enforces a reasonably strict gateway timeout policy. This means that, if you're trying to access a reasonably large dataset, you will need to filter it on the server side using the `datakey`. You might like to review the data manually via the [ABS website](#) to figure out what subset of the data you require.

Note, furthermore, that the datastructure contains a complete codebook for the variables appearing in the relevant dataflow. Since some variables are shared across multiple dataflows, this means that the datastructure corresponding to a particular `id` may contain values for a given variable which are not in the corresponding dataflow.

Value

A data.frame

Examples

```
## Not run:
# List available dataflows
read_api_dataflows()

# Say we want the "Estimated resident population, Country of birth"
# data flow, with the id ERP_COB. We load the data like this:
# Get full data set for a given flow by providing id and start period:
read_api("ERP_COB", start_period = 2020)

# In some cases, loading a whole dataflow (as above) won't work.
# For eg., the `ABS_C16_T10_SA` dataflow is very large,
# so the gateway will timeout if we try to collect the full data set
try(read_api("ABS_C16_T10_SA"))

# We need to filter the dataflow before downloading it.
# To figure out how to filter it, we get metadata ('datastructure').
ds <- read_api_datastructure("ABS_C16_T10_SA")

# The `asgs_2016` code for 'Australia' is 0
ds[ds$var == "asgs_2016" & ds$label == "Australia", ]

# The `sex_abs` code for 'Persons' (i.e. all persons) is 3
ds[ds$var == "sex_abs" & ds$label == "Persons", ]

# So we have:
x <- read_api("ABS_C16_T10_SA", datakey = list(asgs_2016 = 0, sex_abs = 3))
unique(x["asgs_2016"]) # Confirming only 'Australia' level records came through
unique(x["sex_abs"]) # Confirming only 'Persons' level records came through

# Please note however that not all values in the datastructure necessarily
# appear in the data. You get 404s in this case
ds[ds$var == "regiontype" & ds$label == "Destination Zones", ]
try(read_api("ABS_C16_T10_SA", datakey = list(regiontype = "DZN"))))

# If you already have a query url, then use `read_api_url()`
wpi_url <- "https://data.api.abs.gov.au/rest/data/ABS,WPI/all"
read_api_url(wpi_url)

## End(Not run)
```

download_abs_data_cube

Experimental helper function to download ABS data cubes that are not compatible with read_abs.

Description

[Experimental] download_abs_data_cube() downloads the latest ABS data cubes based on the catalogue name (from the website url) and cube. The function downloads the file to disk.

Unlike read_abs(), this function doesn't import or tidy the data. Convenience functions are provided to import and tidy key data cubes; see ?read_payrolls() and ?read_lfs_grossflows().

Usage

```
download_abs_data_cube(
  catalogue_string,
  cube,
  path = Sys.getenv("R_READABS_PATH", unset = tempdir())
)
```

Arguments

catalogue_string	ABS catalogue name as a string from the ABS website. For example, Labour Force, Australia, Detailed is "labour-force-australia-detailed". The possible catalogues can be obtained using the helper function show_available_catalogues(); or search these catalogues using search_catalogues(),
cube	character. A character string that is either the complete filename or (uniquely) in the filename of the data cube you want to download, e.g. "EQ09". The available filenames can be obtained using the helper function get_available_files()
path	Local directory in which downloaded files should be stored. By default, path takes the value set in the environment variable "R_READABS_PATH". If this variable is not set, any files downloaded will be stored in a temporary directory (tempdir()). See Details below for more information.

Details

download_abs_data_cube() downloads an Excel spreadsheet from the ABS.

The file need to be saved somewhere on your disk. This local directory can be controlled using the path argument to read_abs(). If the path argument is not set, read_abs() will store the files in a directory set in the "R_READABS_PATH" environment variable. If this variable isn't set, files will be saved in a temporary directory.

To check the value of the "R_READABS_PATH" variable, run Sys.getenv("R_READABS_PATH"). You can set the value of this variable for a single session using Sys.setenv(R_READABS_PATH = <path>). If you would like to change this variable for all future R sessions, edit your .Renv file and add R_READABS_PATH = <path> line. The easiest way to edit this file is using usethis::edit_r_environ().

The filepath is returned invisibly which enables piping to unzip() or readxl::read_excel.

See Also

Other data cube functions: [search_catalogues\(\)](#), [show_available_catalogues\(\)](#), [show_available_files\(\)](#)

Examples

```
## Not run:
download_abs_data_cube(
  catalogue_string = "labour-force-australia-detailed",
  cube = "EQ09"
)

## End(Not run)
```

extract_abs_sheets	<i>Extract data sheets from an ABS timeseries workbook saved locally as an Excel file.</i>
--------------------	--

Description

Note that this function will not tidy the data for you. Use `read_abs_local()` to import and tidy data from local ABS time series spreadsheets or `read_abs()` to download, import and tidy ABS time series.

Usage

```
extract_abs_sheets(
  filename,
  table_title = NULL,
  path = Sys.getenv("R_READABS_PATH", unset = tempdir())
)
```

Arguments

filename	Filename for an ABS time series spreadsheet (as string)
table_title	String giving the full title of the ABS table, such as "Table 1. Employed persons, Australia"
path	Local directory in which an ABS time series is stored. Default is <code>Sys.getenv("R_READABS_PATH", unset = tempdir())</code> .

```
get_available_lfs_cubes
```

Show the available Labour Force, Australia, detailed data cubes that can be downloaded

Description

Show the available Labour Force, Australia, detailed data cubes that can be downloaded

Usage

```
get_available_lfs_cubes()
```

Details

Intended to be used with `read_lfs_datacube()`. Call `read_lfs_datacube()` interactively, find the table of interest (eg. "LM1"), then use `read_lfs_datacube()`.

Examples

```
get_available_lfs_cubes()
```

```
read_abs
```

Download, extract, and tidy ABS time series spreadsheets

Description

[Stable]

`read_abs()` downloads ABS time series spreadsheets, then extracts the data from those spreadsheets, then tidies the data. The result is a single data frame (tibble) containing tidied data.

Usage

```
read_abs(
  cat_no = NULL,
  tables = "all",
  series_id = NULL,
  path = Sys.getenv("R_READABS_PATH", unset = tempdir()),
  metadata = TRUE,
  show_progressBars = TRUE,
  retain_files = TRUE,
  check_local = TRUE,
  release_date = "latest"
)

read_abs_series(series_id, ...)
```


Arguments

cat_no	ABS catalogue number, as a string, including the extension. For example, "6202.0".
tables	numeric. Time series tables in cat_no`` to download and extract. Default is "all", which will be used to download and import specific tables(s) - eg. tables = 1 or tables = c(1, 5)'. blesto download and import specific tables(s) - eg. tables = 1 or tables = c(1, 5)'.
series_id	(optional) character. Supply an ABS unique time series identifier (such as "A2325807L") to get only that series. This is an alternative to specifying cat_no.
path	Local directory in which downloaded ABS time series spreadsheets should be stored. By default, path takes the value set in the environment variable "R_READABS_PATH". If this variable is not set, any files downloaded by read_abs() will be stored in a temporary directory (tempdir()). See Details below for more information.
metadata	logical. If TRUE (the default), a tidy data frame including ABS metadata (series name, table name, etc.) is included in the output. If FALSE, metadata is dropped.
show_progressBars	TRUE by default. If set to FALSE, progress bars will not be shown when ABS spreadsheets are downloading.
retain_files	when TRUE (the default), the spreadsheets downloaded from the ABS website will be saved in the directory specified with path. If set to FALSE, the files will be stored in a temporary directory.
check_local	If TRUE, the default, local fst files are used, if present.
release_date	Either "latest" or a string coercible to a date, such as "2022-02-01". If "latest", the latest release of the requested data will be returned. If a date, (eg. "2022-02-01") read_abs() will attempt to download the data from that month's release. Note that this only works consistently as expected for monthly data. See Details.
...	Arguments to read_abs_series() are passed to read_abs().

Details

read_abs_series() is a wrapper around read_abs(), with series_id as the first argument.

read_abs() downloads spreadsheet(s) from the ABS containing time series data. These files need to be saved somewhere on your disk. This local directory can be controlled using the path argument to read_abs(). If the path argument is not set, read_abs() will store the files in a directory set in the "R_READABS_PATH" environment variable. If this variable isn't set, files will be saved in a temporary directory.

To check the value of the "R_READABS_PATH" variable, run Sys.getenv("R_READABS_PATH"). You can set the value of this variable for a single session using Sys.setenv(R_READABS_PATH = <path>). If you would like to change this variable for all future R sessions, edit your .Renv file and add R_READABS_PATH = <path> line. The easiest way to edit this file is using use_this::edit_r_environ().

Certain corporate networks restrict your ability to download files in an R session. On some of these networks, the "wininet" method must be used when downloading files. Users can now specify the method that will be used to download files by setting the "R_READABS_DL_METHOD" environment variable.

For example, the following code sets the environment variable for your current session: `sSys.setenv("R_READABS_DL_METHOD = "wininet")` You can add `R_READABS_DL_METHOD = "wininet"` to your `.Renviron` to have this persist across sessions.

The `release_date` argument allows you to download table(s) other than the latest release. This is useful for examining revisions to time series, or for obtaining the version of series that were available on a given date. Note that you cannot supply more than one date to `release_date`. Note also that any dates prior to mid-2019 (the exact date varies by series) will fail. Specifying `release_date` only reliably works for monthly, and some quarterly, data. It does not work for annual data.

Value

A data frame (tibble) containing the tidied data from the ABS time series table(s).

Examples

```
# Download and tidy all time series spreadsheets
# from the Wage Price Index (6345.0)
## Not run:
wpi <- read_abs("6345.0")

## End(Not run)

# Download table 1 from the Wage Price Index
## Not run:
wpi_t1 <- read_abs("6345.0", tables = "1")

## End(Not run)

# Or table 1 as in the Sep 2019 release of the WPI:
## Not run:
wpi_t1_sep2019 <- read_abs("6345.0", tables = "1", release_date = "2019-09-01")

## End(Not run)

# Or tables 1 and 2a from the WPI
## Not run:
wpi_t1_t2a <- read_abs("6345.0", tables = c("1", "2a"))

## End(Not run)

# Get two specific time series, based on their time series IDs
## Not run:
cpi <- read_abs(series_id = c("A2325806K", "A2325807L"))

## End(Not run)

# Get series IDs using the `read_abs_series()` wrapper function
## Not run:
cpi <- read_abs_series(c("A2325806K", "A2325807L"))
```

```
## End(Not run)
```

read_abs_data	<i>Extracts ABS time series data from local Excel spreadsheets and converts to long format.</i>
---------------	---

Description

read_abs_data() is soft deprecated and will be removed in a future version. Please use read_abs_local() to import and tidy locally-stored ABS time series spreadsheets, or read_abs() to download, import, and tidy time series spreadsheets from the ABS website.

Usage

```
read_abs_data(path, sheet)
```

Arguments

path	Filepath to Excel spreadsheet.
sheet	Sheet name or number.

Value

Long-format dataframe

read_abs_local	<i>Read and tidy locally-saved ABS time series spreadsheet(s)</i>
----------------	---

Description

If you need to download and tidy time series data from the ABS, use read_abs(). read_abs_local() imports and tidies data from ABS time series spreadsheets that are already saved to your local drive.

Usage

```
read_abs_local(
  cat_no = NULL,
  filenames = NULL,
  path = Sys.getenv("R_READABS_PATH", unset = tempdir()),
  use_fst = TRUE,
  metadata = TRUE
)
```

Arguments

cat_no	character; a single catalogue number such as "6202.0". When cat_no is specified, all local files in path corresponding to the specified catalogue number will be imported. For example, if you run <code>read_abs_local("6202.0")</code> , it will look in the 6202.0 sub-folder of path and attempt to load any .xls and .xlsx files in that location. If cat_no is specified, filenames will be ignored.
filenames	character vector of at least one filename of a locally-stored ABS time series spreadsheet. For example, "6202001.xls" or <code>c("6202001.xls", "6202005.xls")</code> . Ignored if a value is supplied to cat_no. If filenames is blank and cat_no is blank, <code>read_abs_local()</code> will attempt to read all .xls and .xlsx files in the directory specified with path.
path	path to local directory containing ABS time series file(s). Default is <code>Sys.getenv("R_READABS_PATH", unset = tempdir())</code> . If nothing is specified in filenames or cat_no, <code>read_abs_local()</code> will attempt to read all .xls and .xlsx files in the directory specified with path.
use_fst	logical. If TRUE (the default) then, if an fst file of the tidy data frame has already been saved in path, it is read immediately.
metadata	logical. If TRUE (the default), a tidy data frame including ABS metadata (series name, table name, etc.) is included in the output. If FALSE, metadata is dropped.

Details

Unlike `read_abs()`, the `table_title` column in the data frame returned by `read_abs_local()` is blank. If you require `table_title`, please use `read_abs()` instead.

Examples

```
# Load and tidy two specified files from the "data/ABS" subdirectory
# of your working directory
## Not run:
lfs <- read_abs_local(c("6202001.xls", "6202005.xls"))

## End(Not run)
```

read_abs_metadata	<i>Extracts ABS series metadata directly from Excel spreadsheets and converts to long-form.</i>
-------------------	---

Description

Extracts ABS series metadata directly from Excel spreadsheets and converts to long-form.

Usage

```
read_abs_metadata(path, sheet)
```

Arguments

path	Filepath to Excel spreadsheet.
sheet	Sheet name or number.

Value

Long-form dataframe

read_abs_url	<i>Download and import an ABS time series spreadsheet from a given URL</i>
--------------	--

Description

Download and import an ABS time series spreadsheet from a given URL

Usage

```
read_abs_url(
  url,
  path = Sys.getenv("R_READABS_PATH", unset = tempdir()),
  show_progressBars = TRUE,
  ...
)
```

Arguments

url	Character vector of url(s) to ABS time series spreadsheet(s).
path	Local directory in which downloaded ABS time series spreadsheets should be stored. By default, path takes the value set in the environment variable "R_READABS_PATH". If this variable is not set, any files downloaded by read_abs() will be stored in a temporary directory (tempdir()). See ?read_abs() for more.
show_progressBars	TRUE by default. If set to FALSE, progress bars will not be shown when ABS spreadsheets are downloading.
...	Additional arguments passed to read_abs_local().

Details

If you have a specific URL to the time series spreadsheet you wish to download, read_abs_url() will download, import and tidy it. This is useful for older vintages of data, or discontinued data.

Examples

```
## Not run:
url <- paste0(
  "https://www.abs.gov.au/statistics/labour/",
  "employment-and-unemployment/labour-force-australia/aug-2022/6202001.xlsx"
)
read_abs_url(url)

## End(Not run)
```

read_awe

read_awe

Description

Convenience function to obtain wage levels from ABS 6302.0, Average Weekly Earnings, Australia.

Usage

```
read_awe(
  wage_measure = c("awote", "ftawe", "awe"),
  sex = c("persons", "males", "females"),
  sector = c("total", "private", "public"),
  state = c("all", "nsw", "vic", "qld", "sa", "wa", "tas", "nt", "act"),
  na.rm = FALSE,
  path = Sys.getenv("R_READABS_PATH", unset = tempdir()),
  show_progressBars = FALSE,
  check_local = FALSE
)
```

Arguments

wage_measure	Character of length 1. Must be one of: awote Average weekly ordinary time earnings; also known as Full-time adult ordinary time earnings ftawe Full-time adult total earnings awe Average weekly total earnings of all employees
sex	Character of length 1. Must be one of: persons, males, or females.
sector	Character of length 1. Must be one of: total, private, or public. Note that you cannot get sector-by-state data; if state is not all then sector must be total.
state	Character of length 1. Must be one of: all, nsw, vic, qld, sa, wa, nt, or act. Note that you cannot get sector-by-state data; if sector is not total then state must be all.

na.rm	Logical. FALSE by default. If FALSE, a consistent quarterly series is returned, with NA values for quarters in which there is no data. If TRUE, only dates with data are included in the returned data frame.
path	See ?read_abs
show_progress_bars	See ?read_abs
check_local	See ?read_abs

Details

The latest AWE data is available using `read_abs(cat_no = "6302.0", tables = 2)`. However, this time series only goes back to 2012, when the ABS switched from quarterly to biannual collection and release of the AWE data. The `read_awe()` function assembles on time series back to November 1983 quarter; it is quarterly to 2012 and biannual from then. Note that the data returned with this function is consistently quarterly; any quarters for which there are no observations are recorded as NA unless `na.rm = TRUE`.

Value

A `tbl_df` with four columns: `date`, `sex`, `wage_measure` and `value`. The data is nominal and seasonally adjusted.

Examples

```
## Not run:
read_awe("awote", "persons")

## End(Not run)
```

read_cpi	<i>Download a tidy tibble containing the Consumer Price Index from the ABS</i>
----------	--

Description

`read_cpi()` uses the `read_abs()` function to download, import, and tidy the Consumer Price Index from the ABS. It returns a tibble containing two columns: the date and the CPI index value that corresponds to that date. This makes joining the CPI to another dataframe easy. `read_cpi()` returns the original (ie. not seasonally adjusted) all groups CPI for Australia. If you want the analytical series (eg. seasonally adjusted CPI, or trimmed mean CPI), you can use `read_abs()`.

Usage

```
read_cpi(
  path = Sys.getenv("R_READABS_PATH", unset = tempdir()),
  show_progressBars = TRUE,
  check_local = FALSE,
  retain_files = FALSE
)
```

Arguments

`path` character; default is "data/ABS". Only used if `retain_files` is set to `TRUE`. Local directory in which to save downloaded ABS time series spreadsheets.

`show_progressBars` logical; `TRUE` by default. If set to `FALSE`, progress bars will not be shown when ABS spreadsheets are downloading.

`check_local` logical; `FALSE` by default. See `?read_abs`.

`retain_files` logical; `FALSE` by default. When `TRUE`, the spreadsheets downloaded from the ABS website will be saved in the directory specified with `'path'`.

Examples

```
# Create a tibble called 'cpi' that contains the CPI index
# numbers for each quarter

cpi <- read_cpi()

# This tibble can now be joined to another to help streamline the process of
# deflating nominal values.
```

read_erp	<i>Download a tidy tibble containing the Estimated Residential Population from the ABS</i>
----------	--

Description

`read_erp()` uses the `read_abs()` function to download, import, and tidy the Estimated Residential Population from the ABS. It allows the user to specify age, sex and states/territories of interest. It returns a tibble containing five columns: the date, the age range, sex and states that the ERP corresponds to. This makes joining the ERP to another dataframe easy.

Usage

```
read_erp(
  age_range = 0:100,
  sex = "Persons",
  states = c("Australia", "New South Wales", "Victoria", "Queensland", "South Australia"),
```



```

    "Western Australia", "Tasmania", "Northern Territory",
    "Australian Capital Territory"),
  path = Sys.getenv("R_READABS_PATH", unset = tempdir()),
  show_progressBars = TRUE,
  check_local = FALSE,
  retain_files = FALSE
)

```

Arguments

age_range	numeric; default is "0:100". A vector containing ages in single years for which an ERP is sought. The ABS top-code ages at 100.
sex	character; default is "Persons". Other values are "Male" and "Female". Multiple values allowed.
states	character; default is "Australia". Other values are the full or abbreviated names of the states and self-governing territories. Multiple values allowed.
path	character; default is "data/ABS". Only used if retain_files is set to TRUE. Local directory in which to save downloaded ABS time series spreadsheets.
show_progressBars	logical; TRUE by default. If set to FALSE, progress bars will not be shown when ABS spreadsheets are downloading.
check_local	logical; FALSE by default. See ?read_abs.
retain_files	logical; FALSE by default. When TRUE, the spreadsheets downloaded from the ABS website will be saved in the directory specified with 'path'.

Examples

```

# Create a tibble called 'erp' that contains the ERP index
# numbers for 30 June each year for Australia.

erp <- read_erp()

```

read_job_mobility *Download and tidy ABS Job Mobility tables*

Description

Import a tidy tibble of ABS Job Mobility data

Usage

```

read_job_mobility(
  tables = "all",
  path = Sys.getenv("R_READABS_PATH", unset = tempdir())
)

```

Arguments

tables	Either "all" (the default) to import all tables, or a vector of table numbers, such as 1 or c(2, 4).
path	Local directory in which downloaded ABS time series spreadsheets should be stored. By default, 'path' takes the value set in the environment variable "R_READABS_PATH". If this variable is not set, any files downloaded by read_abs() will be stored in a temporary directory (tempdir()).

Examples

```
## Not run:
# Get all tables from the ABS Job Mobility series
read_job_mobility()

# Get tables 1 and 2
read_job_mobility(c(1, 2))

## End(Not run)
```

read_lfs_datacube	<i>Convenience function to download and tidy data cubes from ABS Labour Force, Australia, Detailed.</i>
-------------------	---

Description

Convenience function to download and tidy data cubes from ABS Labour Force, Australia, Detailed.

Usage

```
read_lfs_datacube(cube, path = Sys.getenv("R_READABS_PATH", unset = tempdir()))
```

Arguments

cube	character. A character string that is either the complete filename or (uniquely) in the filename of the data cube you want to download. Use get_available_lfs_cubes() to see a dataframe of options.
path	Local directory in which downloaded files should be stored.

Value

A tibble with the data from the data cube. Columns names are tidied and dates are converted to Date class.

Examples

```
read_lfs_datacube("EQ02")
```

read_lfs_grossflows	<i>Download, import and tidy 'gross flows' data cube from the monthly ABS Labour Force survey.</i>
---------------------	--

Description

This convenience function downloads, imports and tidies the 'gross flows' data cube from the monthly ABS Labour Force survey. The gross flows data cube (GM1) shows estimates of the number of people who transitioned from one labour force status to another between two months.

Usage

```
read_lfs_grossflows(  
  weights = c("current", "previous"),  
  path = Sys.getenv("R_READABS_PATH", unset = tempdir())  
)
```

Arguments

weights	either "current" or "previous". If "current", figures will use the current month's Labour Force survey weights; if "previous", the previous month's weights are used.
path	Local directory in which downloaded files should be stored. By default, 'path' takes the value set in the environment variable "R_READABS_PATH". If this variable is not set, any files downloaded will be stored in a temporary directory (tempdir()). See Details in ?read_abs for more information.

Value

A tibble containing data cube GM1 from the monthly Labour Force survey.

Examples

```
## Not run:  
read_lfs_grossflows()  
  
## End(Not run)
```

read_payrolls	<i>Download and tidy ABS payroll jobs and wages data</i>
---------------	--

Description

Import a tidy tibble of ABS Payroll Jobs data.

Usage

```
read_payrolls(
  series = c("industry_jobs", "subindustry_jobs", "empsize_jobs"),
  path = Sys.getenv("R_READABS_PATH", unset = tempdir())
)
```

Arguments

series	<p>Character. Must be one of:</p> <p>"industry_jobs" Payroll jobs by industry division, state, and age group (Table 1)</p> <p>"subindustry_jobs" Payroll jobs by industry sub-division and industry division (Table 2)</p> <p>"empsize_jobs" Payroll jobs by size of employer (number of employees) and state/territory (Table 3)</p> <p>The default is "industry_jobs".</p>
path	<p>Local directory in which downloaded ABS time series spreadsheets should be stored. By default, path takes the value set in the environment variable "R_READABS_PATH". If this variable is not set, any files downloaded by read_abs() will be stored in a temporary directory (tempdir()).</p>

Details

The ABS **Payroll Jobs** dataset draws upon data collected by the Australian Taxation Office as part of its Single-Touch Payroll initiative and supplements the monthly Labour Force Survey. Unfortunately, the data as published by the ABS (1) is not in a standard time series spreadsheet; and (2) is messy in various ways that make it hard to read in R. This convenience function uses `download_abs_data_cube()` to import the payrolls data, and then tidies it up.

Note that this ABS release used to be called Weekly Payroll Jobs and Wages Australia. The total wages series were removed from this release in mid-2023 and it was renamed to Weekly Payroll Jobs. The ability to read total wages indexes using this function was therefore also removed. It was then renamed Payroll Jobs and the frequency was reduced, with further modifications to the data released.

Value

A tidy (long) `tbl_df`. The number of columns differs based on the series.

Examples

```
## Not run:
# Fetch payroll jobs by industry and state (the default, "industry_jobs")
read_payrolls()

# Payroll jobs by employer size
read_payrolls("empsize_jobs")

## End(Not run)
```

scrape_abs_catalogues *Helper function for download_abs_data_cube to scrape the available catalogues from the ABS website.*

Description

This function downloads a new version of the lookup table used by show_available_catalogues.

Usage

```
scrape_abs_catalogues()
```

Value

A tibble containing the catalogues and how they are organised on the ABS website.

search_catalogues *Search for ABS catalogues that match a string*

Description

[Experimental] Helper function to use with download_abs_data_cube().

download_abs_data_cube() requires that you specify a catalogue. search_catalogues() helps you find the catalogue you want, by searching for a given string in the catalogue names, product title, and broad topic.

Usage

```
search_catalogues(string, refresh = FALSE)
```

Arguments

string	Character. A word or phrase you want to search for, such as "labour" or "union". Not case sensitive.
refresh	Logical. FALSE by default. If TRUE, will re-scrape the ABS website to ensure that the list of catalogues is up-to-date.

Value

A data frame (tibble) containing the topic (heading), product title (sub_heading), catalogue (catalogue) and URL (URL) of any catalogues that match the provided string.

See Also

Other data cube functions: [download_abs_data_cube\(\)](#), [show_available_catalogues\(\)](#), [show_available_files\(\)](#)

Examples

```
search_catalogues("labour")
```

```
search_files
```

Search for a file within an ABS catalogue

Description

Search for a file within an ABS catalogue

Usage

```
search_files(string, catalogue, refresh = FALSE)
```

Arguments

string	String to search for among filenames in a catalogue
catalogue	Name of catalogue
refresh	logical; FALSE by default. When TRUE, will re-scrape the list of files within the catalogue.

Examples

```
## Not run:
search_files("GM1", "labour-force-australia")

## End(Not run)
```

separate_series	<i>Separate the series column in a tidy ABS time series data frame</i>
-----------------	--

Description

Separate the 'series' column in a data frame (tibble) downloaded using `read_abs()` into multiple columns using the ";" separator.

Usage

```
separate_series(  
  data,  
  column_names = NULL,  
  remove_totals = FALSE,  
  remove_nas = FALSE  
)
```

Arguments

<code>data</code>	A data frame (tibble) containing tidied data from the ABS time series table(s).
<code>column_names</code>	(optional) character vector. Supply a vector of column names, such as <code>c("group_name", "variable", "gender")</code> . If not supplied, columns will be named "series_1" etc.
<code>remove_totals</code>	logical. FALSE by default. If set to TRUE, any series rows that contain the word "total" will be removed.
<code>remove_nas</code>	logical. FALSE by default. If set to TRUE, any rows containing an NA in at least one of the separated series columns will be removed.

Value

A data frame (tibble) containing the tidied data from the ABS time series table(s).

Examples

```
## Not run:  
wpi <- read_abs("6345.0", 1) %>%  
  separate_series()  
  
## End(Not run)
```

show_available_catalogues

Helper function for download_abs_data_cube to show the available catalogues.

Description

[Experimental]

This function lists the possible catalogues that are available on the ABS website. These catalogues must be specified as a string as an argument to download_abs_data_cube.

Usage

```
show_available_catalogues(selected_heading = NULL, refresh = FALSE)
```

Arguments

selected_heading

optional character string specifying the heading on the [ABS statistics webpage](#). e.g. "Earnings and work hours"

refresh

logical; FALSE by default. If FALSE, an internal table of the available ABS catalogues is used. If TRUE, this table is refreshed from the ABS website.

Value

a character vector of catalogues.

See Also

Other data cube functions: [download_abs_data_cube\(\)](#), [search_catalogues\(\)](#), [show_available_files\(\)](#)

Examples

```
show_available_catalogues("Earnings and work hours")
```

show_available_files *Helper function to show the files available in a particular catalogue number.*

Description

[Experimental] To be used in conjunction with download_abs_data_cube().

This function lists the possible files that are available in a catalogue. The filename (or an unambiguous part of the filename) must be specified as a string as an argument to download_abs_data_cube.

Usage

```
show_available_files(catalogue_string, refresh = FALSE)
```

```
get_available_files(catalogue_string, refresh = FALSE)
```

Arguments

`catalogue_string` character string specifying the catalogue, e.g. "labour-force-australia-detailed". You can use `show_available_catalogues()` see all the possible catalogues, or `search_catalogues()` to find catalogues that contain a given string.

`refresh` logical; FALSE by default. If FALSE, an internal table of the available ABS catalogues is used. If TRUE, this table is refreshed from the ABS website.

Details

`get_available_files()` is an alias for `show_available_files()`.

Value

A tibble containing the title of the file, the filename and the complete url.

See Also

Other data cube functions: [download_abs_data_cube\(\)](#), [search_catalogues\(\)](#), [show_available_catalogues\(\)](#)

Other data cube functions: [download_abs_data_cube\(\)](#), [search_catalogues\(\)](#), [show_available_catalogues\(\)](#)

Examples

```
## Not run:
show_available_files("labour-force-australia-detailed")

## End(Not run)
```

tidy_abs

Tidy ABS time series data.

Description

Tidy ABS time series data.

Usage

```
tidy_abs(df, metadata = TRUE)
```

Arguments

df	A data frame containing ABS time series data that has been extracted using <code>extract_abs_sheets</code> .
metadata	logical. If TRUE (the default), a tidy data frame including ABS metadata (series name, table name, etc.) is included in the output. If FALSE, metadata is dropped.

Value

data frame (tibble) in long format.

Examples

```
# First extract the data from the local spreadsheet
## Not run:
wpi <- extract_abs_sheets("634501.xls")

## End(Not run)

# Then tidy the data extracted from the spreadsheet. Note that
# \code{extract_abs_sheets()} returns a list of data frames, so we need to
# subset the list.
## Not run:
tidy_wpi <- tidy_abs(wpi[[1]])

## End(Not run)
```

tidy_abs_list

Tidy multiple dataframes of ABS time series data contained in a list.

Description

Tidy multiple dataframes of ABS time series data contained in a list.

Usage

```
tidy_abs_list(list_of_dfs, metadata = TRUE)
```

Arguments

list_of_dfs	A list of dataframes containing extracted ABS time series data.
metadata	logical. If TRUE (the default), a tidy data frame including ABS metadata (series name, table name, etc.) is included in the output. If FALSE, metadata is dropped.

Index

* data cube functions

- download_abs_data_cube, 6
- search_catalogues, 21
- show_available_catalogues, 24
- show_available_files, 24

abs_api, 2

check_latest_date, 4

download_abs_data_cube, 6, 22, 24, 25

extract_abs_sheets, 7

get_available_files
(show_available_files), 24

get_available_lfs_cubes, 8

read_abs, 8

read_abs_data, 11

read_abs_local, 11

read_abs_metadata, 12

read_abs_series (read_abs), 8

read_abs_url, 13

read_api (abs_api), 2

read_api_dataflows (abs_api), 2

read_api_datastructure (abs_api), 2

read_api_url (abs_api), 2

read_awe, 14

read_cpi, 15

read_erp, 16

read_job_mobility, 17

read_lfs_datacube, 18

read_lfs_grossflows, 19

read_payrolls, 20

scrape_abs_catalogues, 21

search_catalogues, 7, 21, 24, 25

search_files, 22

separate_series, 23

show_available_catalogues, 7, 22, 24, 25

show_available_files, 7, 22, 24, 24

tidy_abs, 25

tidy_abs_list, 26