Package 'cancensus'

January 23, 2023

Type Package

Title Access, Retrieve, and Work with Canadian Census Data and Geography

Version 0.5.5

Description Integrated, convenient, and uniform access to Canadian
Census data and geography retrieved using the 'CensusMapper' API. This package produces analysis-ready
tidy data frames and spatial data in multiple formats, as well as convenience functions for working with Census variables, variable hierarchies, and region selection. API keys are freely available with free registration at <https://censusmapper.ca/api>. Census data and boundary geometries are reproduced and distributed on an ``as is'' basis with the permission of Statistics Canada (Statistics Canada 2001; 2006; 2011; 2016; 2021).

License MIT + file LICENSE

Encoding UTF-8

LazyData yes

ByteCompile yes

NeedsCompilation no

Imports digest (>= 0.1), dplyr (>= 0.7), httr (>= 1.0.0), jsonlite (>= 1.0), rlang

RoxygenNote 7.2.1

Suggests knitr, ggplot2, leaflet, mapdeck, rmarkdown, readr, rgdal, rgeos, scales, sp, sf, geojsonsf, tidyr, lwgeom, xml2

VignetteBuilder knitr

URL https://github.com/mountainMath/cancensus,
https://mountainmath.github.io/cancensus/,
https://censusmapper.ca/api

BugReports https://github.com/mountainMath/cancensus/issues

Depends R (>= 2.10)

Author Jens von Bergmann [aut] (API creator and maintainer),
Dmitry Shkolnik [aut, cre] (Package maintainer, responsible for correspondence),
Aaron Jacobs [aut]

Maintainer Dmitry Shkolnik <shkolnikd@gmail.com>

Repository CRAN

Date/Publication 2023-01-23 08:40:06 UTC

R topics documented:

add_unique_names_to_region_list

Convenience function for creating unique names from region list

Description

Names of municipalities are not always unique, especially at the CSD level. This function takes as input a subset of a regions list as generated from 'list_census_regions()' and de-duplicates names as needed by adding the municipal status in parenthesis. If this does not de-duplicate the name then the geographic identifier will be further added in parenthesis behind that.

Usage

```
add_unique_names_to_region_list(region_list)
```

Arguments

region_list	a subset of a regions list as gotten from 'list_census_regions()'

Value

The same list of regions with an extra column 'Name' with de-duplicated names.

Examples

```
## Not run:
# This will return a warning that no match was found, but will suggest similar named regions.
library(dplyr)
list_census_regions("CA21") %>%
  filter(level=="CSD", CMA_UID=="59933") %>%
  add_unique_names_to_region_list()

## End(Not run)
```

as_census_region_list

Convert a (suitably filtered) data frame from list_census_regions to a list suitable for passing to get_census.

Description

Convert a (suitably filtered) data frame from list_census_regions to a list suitable for passing to get_census.

Usage

```
as_census_region_list(tbl)
```

Arguments

tbl
: A data frame, suitably filtered, as returned by `list_census_regions`.

Examples

```
## Not run:
library(dplyr, warn.conflicts = FALSE)

# Query the CensusMapper API for the total occupied dwellings
# of 20 random Census Subdivisions, in Census 2016.
regions <- list_census_regions("CA16") %>%
  filter(level == "CSD") %>%
  sample_n(20) %>%
  as_census_region_list()

occupied <- get_census("CA16", regions = regions,
                            vectors = c("v_CA16_408"),
                            level = "Regions")

## End(Not run)
```

---

child_census_vectors — *List all child variables from vector hierarchies given either a list of Census variables returned by* `list_census_vectors`, `search_census_vectors`, `find_census_vectors`*, or a direct string reference to the vector code.*

Description

List all child variables from vector hierarchies given either a list of Census variables returned by `list_census_vectors`, `search_census_vectors`, `find_census_vectors`, or a direct string reference to the vector code.

Usage

```
child_census_vectors(
  vector_list,
  leaves_only = FALSE,
  max_level = NA,
  keep_parent = FALSE
)
```

Arguments

vector_list
: the list of vectors to be used, either a character vector or a filtered tibble as returned from `list_census_vectors`.

leaves_only
: boolean flag to indicate if only final leaf vectors should be returned, i.e. terminal vectors that themselves do not have children.

`max_level`	optional, maximum depth to look for child vectors. Default is `NA` and will return all child census vectors.
`keep_parent`	optional, also return parent vector in list of results. Default is set to `FALSE`.

Examples

```
# Query parent vectors directly using vector identifier
child_census_vectors("v_CA16_2510")

## Not run:

# Example using multiple vectors coerced into a list
child_census_vectors(c("v_CA16_2510","v_CA16_2511","v_CA16_2512"))

# or, equivalently
selected_vectors <- c("v_CA16_2510","v_CA16_2511","v_CA16_2512")
child_census_vectors(selected_vectors)

# Example using dplyr and piped arguments
library(dplyr, warn.conflicts = FALSE)

list_census_vectors("CA16") %>%
  filter(vector == "v_CA16_2510") %>%
  child_census_vectors(TRUE)

# this will return the equivalent of c("v_CA16_2510", child_census_vectors("v_CA16_2510"))
list_census_vectors("CA16") %>%
  filter(vector == "v_CA16_2510") %>%
  child_census_vectors(TRUE, keep_parent = TRUE)

## End(Not run)
```

`CODES_TABLE` *A dataset with code table summaries for census data*

Description

A dataset with code table summaries for census data

Author(s)

derived from StatCan definitions

References

https://www12.statcan.gc.ca/census-recensement/2021/geo/ref/domain-domaine/index2021-eng.cfm?lang=e&id=CSDtype, https://www12.statcan.gc.ca/census-recensement/2021/geo/ref/domain-domaine/index2021-eng.cfm?lang=e&id=CDtype

COV_SKYTRAIN_STATIONS *A dataset City of Vancouver skytrain station locations*

Description

A dataset City of Vancouver skytrain station locations

Author(s)

City of Vancouver Open Data

References

https://opendata.vancouver.ca/explore/dataset/rapid-transit-stations/information/

dataset_attribution *Get attribution for datasets*

Description

Get attribution for datasets

Usage

```
dataset_attribution(datasets)
```

Arguments

datasets	Vector of dataset identifiers

Value

Returns a string to be used as attribution for the given datasets.

Examples

```
# Attribution string for the 2006 and 2016 census datasets
dataset_attribution(c('CA06','CA16'))
```

explore_census_regions *Interactively browse Census variables and regions on Censusmapper.ca in a new browser window*

Description

Finding the right Census variables or regions can be complicated. `explore_census_vectors(dataset)` and `explore_census_regions(dataset)` will open a new browser page or tab to an interactive Census variable and region exploration and selection tool on the Censusmapper.ca website. Interactive tools available for the CA16, CA11, CA06, and CA01 Census datasets and geographies.

Usage

```
explore_census_regions(dataset = "CA16")
```

Arguments

`dataset`	The dataset to query for available vectors, e.g. `'CA16'`. Interactive tools available for the CA16, CA11, CA06, and CA01 Census datasets and geographies.

Examples

```
## Not run:

explore_census_vectors(dataset = "CA16")

explore_census_regions(dataset = "CA11")


## End(Not run)
```

explore_census_vectors *Interactively browse Census variables and regions on Censusmapper.ca in a new browser window*

Description

Finding the right Census variables or regions can be complicated. `explore_census_vectors(dataset)` and `explore_census_regions(dataset)` will open a new browser page or tab to an interactive Census variable and region exploration and selection tool on the Censusmapper.ca website. Interactive tools available for the CA16, CA11, CA06, and CA01 Census datasets and geographies.

Usage

```
explore_census_vectors(dataset = "CA16")
```

Arguments

dataset — The dataset to query for available vectors, e.g. 'CA16'. Interactive tools available for the CA16, CA11, CA06, and CA01 Census datasets and geographies.

Examples

```
## Not run:

explore_census_vectors(dataset = "CA16")

explore_census_regions(dataset = "CA11")


## End(Not run)
```

---

find_census_vectors — *Query the CensusMapper API for vectors using exact, semantic, or keyword search*

Description

Query the available list of Census vectors based on their label and return details including vector code. Default search behaviour expects an exact match, but keyword or semantic searches can be used instead by setting `query_type='keyword'` or `query_type = 'semantic'` instead. Keyword search is useful when looking to explore Census vectors based on broad themes like "income" or "language". Keyword search separates the query into unigrams and returns Census vectors with matching words, ranked by incidence of matches. Semantic search is designed for more precise searches while allowing room for error for spelling or phrasing, as well as for finding closely related vector matches. Semantic search separates the query into n-grams and relies on string distance measurement using a generalized Levenshtein distance approach.

Some census vectors return population counts segmented by `Female` and `Male` populations, in addition to a total aggregate. By default, query matches will return matches for the `Total` aggregation, but can optionally return only the `Female` or `Male` aggregations by adding `type = 'female'` or `type = 'male'` as a parameter.

Usage

```
find_census_vectors(query, dataset, type = "all", query_type = "exact", ...)
```

Arguments

query — The term or phrase to search for e.g. 'Oji-cree'. Search queries are case insensitive.

dataset — The dataset to query for available vectors, e.g. 'CA16'. To see a list of available datasets: `list_census_datasets()`

type — One of 'all', 'total', 'male' or 'female'. If specified, only return aggregations of specified ‘type‘. By default, only the 'total' aggregation will be returned.

query_type — One of exact, 'semantic' or 'keyword'. By default, assumes exact string matching, but the alternatives may be better options in some cases. See description section for more details on query types.

... — Other arguments passed to internal functions.

Examples

```
find_census_vectors('Oji-cree', dataset = 'CA16', type = 'total', query_type = 'exact')

find_census_vectors('commuting duration', dataset = 'CA11', type = 'female', query_type = 'keyword')

find_census_vectors('after tax income', dataset = 'CA16', type = 'total', query_type = 'semantic')

## Not run:
# This incorrect spelling will return a warning that no match was found,
# but will suggest trying semantic or keyword search.
find_census_vectors('Ojibwey', dataset = 'CA16', type = 'total')

# This will find near matches as well
find_census_vectors('Ojibwey', dataset = 'CA16', type = 'total', query_type = "semantic")

find_census_vectors('commute duration', dataset = 'CA16', type = 'female', query_type = 'keyword')

find_census_vectors('commute duration', dataset = 'CA11', type = 'all', query_type = 'keyword')

find_census_vectors('ukrainian origin', dataset = 'CA16', type = 'total', query_type = 'keyword')


## End(Not run)
```

---

get_census — *Access to Canadian census data through the CensusMapper API*

Description

This function allows convenient access to Canadian census data and boundary files through the CensusMapper API. An API key is required to retrieve data.

Usage

```
get_census(
  dataset,
  regions,
  level = NA,
  vectors = c(),
```

```
  geo_format = NA,
  resolution = "simplified",
  labels = "detailed",
  use_cache = TRUE,
  quiet = FALSE,
  api_key = Sys.getenv("CM_API_KEY")
)
```

Arguments

dataset	A CensusMapper dataset identifier.
regions	A named list of census regions to retrieve. Names must be valid census aggregation levels.
level	The census aggregation level to retrieve, defaults to "Regions". One of "Regions", "PR", "CMA", "CD", "CSD", "CT", "DA", "EA" (for 1996), or "DB" (for 2001-2016).
vectors	An R vector containing the CensusMapper variable names of the census variables to download. If no vectors are specified only geographic data will get downloaded.
geo_format	By default is set to NA and appends no geographic information. To include geographic information with census data, specify one of either "sf" to return an sf object (requires the sf package) or "sp" to return a SpatialPolygonsDataFrame-class object (requires the rgdal package). If user requests geo-spatial data and neither package is available, a context menu will prompt to install the sf package.
resolution	Resolution of the geographic data. cancensus will download simplified geometries by default. For lower level geometries like DB or DA this will be very close to the high resolution data. Simplification generally increases as the geographic aggregation level increases. If high resolution geometries are required then this option can be set to 'high'. By default this setting is set to 'simplified'.
labels	Set to "detailed" by default, but truncated Census variable names can be selected by setting labels = "short". Use label_vectors(...) to return variable label information in detail.
use_cache	If set to TRUE (the default) data will be read from the local cache if available.
quiet	When TRUE, suppress messages and warnings.
api_key	An API key for the CensusMapper API. Defaults to options() and then the CM_API_KEY environment variable.

Details

For help selecting regions and vectors, see list_census_regions and list_census_vectors, or check out the interactive selection tool at https://censusmapper.ca/api by calling explore_census_vectors()

Source

Census data and boundary geographies are reproduced and distributed on an "as is" basis with the permission of Statistics Canada (Statistics Canada 1996; 2001; 2006; 2011; 2016).

Examples

```
# Query the API for data on dwellings in Vancouver, at the census subdivision
# level:
## Not run:
census_data <- get_census(dataset='CA16', regions=list(CMA="59933"),
                          vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"),
                          level='CSD')

# Query the API for data on dwellings in Vancouver, at the census subdivision
# level, and return the associated geography files in \code{sf} format:
census_data <- get_census(dataset='CA16', regions=list(CMA="59933"),
                          vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"),
                          level='CSD', geo_format = "sf")

# Make the same query, but this time drop descriptive vector names:
census_data <- get_census(dataset='CA16', regions=list(CMA="59933"),
                          vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"),
                          level='CSD', geo_format = "sf", labels="short")

# Get details for truncated vectors:
label_vectors(census_data)

## End(Not run)
```

---

get_intersecting_geometries

Get identifiers for census regions intersecting a geometry

Description

This function returns a list of regions that intersect a given geometry input. This list of regions can be used directly to query census when one is interested in census data for a particular geographic region that does not coincide with defined census geometries. The returned value can be used as the `regions` parameter in get_census to get corresponding census geographies and variables that cover the give geometry. Input spatial objects can be any `sf` or `sfc` class objects such as `POINT`, `MULTIPOINT` or `POLYGON`.

This function comes with CensusMapper API limits

Usage

```
get_intersecting_geometries(
  dataset,
  level,
  geometry,
  simplified = FALSE,
  use_cache = TRUE,
  quiet = FALSE,
  api_key = Sys.getenv("CM_API_KEY")
)
```

Arguments

`dataset`	A CensusMapper dataset identifier.
`level`	The census aggregation level to retrieve. One of "Regions", "PR", "CMA", "CD", "CSD", "CT", "DA", "EA" (for 1996 census), or "DB" (for 2001-2016 censuses)..
`geometry`	A valid `sf` or `sfc` class object. As long as the geometry is valid, any projection is accepted. Objects will be reprojected as server-side intersections use lat/lon projection.
`simplified`	If `TRUE` will return a region list compatible with get_census, otherwise will return a character vector of matching region ids.
`use_cache`	If set to `TRUE` (the default) data will be read from the local cache if available.
`quiet`	When TRUE, suppress messages and warnings.
`api_key`	An API key for the CensusMapper API. Defaults to `options()` and then the `CM_API_KEY` environment variable.

Source

Census data and boundary geographies are reproduced and distributed on an "as is" basis with the permission of Statistics Canada (Statistics Canada 1996; 2001; 2006; 2011; 2016).

Examples

```
## Not run:
# Example using a POINT-class object from a pair of lat/lon coordinates
point_geo <- sf::st_sfc(sf::st_point(c(-123.25149, 49.27026)), crs=4326)
regions <- get_intersecting_geometries(dataset = 'CA16', level = 'CT', geometry = point_geo)
census_data <- get_census(dataset='CA16', regions=regions,
                          vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"),
                          level='CT')

# Example using a POLYGON-class object from a bounding box with four coordinates
poly_geo <- sf::st_as_sfc(sf::st_bbox(c(ymin = 49.25, ymax = 49.30,
                          xmin = -123.25, xmax = -123.30)), crs = 4326)
regions <- get_intersecting_geometries(dataset = 'CA16', level = 'CT', geometry = poly_geo)
census_data <- get_census(dataset='CA16', regions=regions,
                         vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"), level='CT')


## End(Not run)
```

`get_statcan_geographic_attributes`

Read the Geographic Attributes File

Description

Reads the Geographies Attributes File for the given census year. The table contains the information on how all the geographic levels are related for each area, and population, dwelling and household counts. Data gets cached after first download if the cancensus cache path has been set. A reference guide is available at https://www150.statcan.gc.ca/n1/en/catalogue/92-151-G2021001

Usage

```
get_statcan_geographic_attributes(census_year = "2021", refresh = FALSE)
```

Arguments

`census_year`	census year to get the data for, right now only 2006, 2011, 2016, 2021 are supported
`refresh`	(logical) refresh the cache if true

Value

tibble with the relationship data

Examples

```
# list add the cached census data
## Not run:
get_statcan_geographic_attributes("2021")

## End(Not run)
```

`get_statcan_geographies` *Read the geosuite data*

Description

Reads the original unprocessed geographic boundary files from Statistics Canada

Usage

```
get_statcan_geographies(
  census_year,
  level,
  type = "cartographic",
  cache_path = NULL,
  timeout = 1000,
  refresh = FALSE,
  quiet = FALSE
)
```

Arguments

census_year	census year to get the data for, right now only 2021 is supported
level	geographic level to return the data for, valid choices are "PR","CD","CMACA","CSD","CT","ADA","DA
type	type of geographic data, valid choices area "cartographic" or "digital"
cache_path	optional path to cache the data. If the cancensus cache path is set the geographic data gets cached in the "geographies" subdirectory of the cancensus cache path.
timeout	optional timeout parameter, adjust as needed if the data download times out when using slow connections
refresh	(logical) refresh the cache if true
quiet	(logical) suppress messages if ‘TRUE‘

Value

a spatial dataframe with the geographic data

Examples

```
# get the digital geographic boundaries for provinces and territories
## Not run:
get_statcan_geographies(census_year="2021",level="PR",type="digital")

## End(Not run)
```

---

get_statcan_geo_suite *Read the geosuite data*

Description

Reads the geosuite data for the given level and census year. Data gets cached after first download if the cancensus cache path has been set. For older years ‘get_statcan_geographic_attributes()‘ can fill in most of the information

Usage

```
get_statcan_geo_suite(level, census_year = "2021", refresh = FALSE)
```

Arguments

level	geographic level to return the data for, valid choices are "DB", "DA", "ADA", "CT", "CSD", "CMA", "CD", "PR", "FED", "DPL", "ER", "PN", "POPCTR"
census_year	census year to get the data for, right now only 2021 is supported
refresh	(logical) refresh the cache if true

Value

tibble with the geosuite data

Examples

```
# list add the cached census data
## Not run:
get_statcan_geo_suite("DA","2021")

## End(Not run)
```

`get_statcan_wds_data` *Query the StatCan WDS for data*

Description

Get official census data from Statistics Canada for a given set of DGUIDs. Only available for the 2021 census. The downloaded data gets enriched by geographic and characteristic names based on metadata obtained via ‘get_statcan_wds_metadata()‘. Data is cached for the duration of the R session.

Usage

```
get_statcan_wds_data(
  DGUIDs,
  members = NULL,
  gender = "All",
  language = "en",
  refresh = FALSE
)
```

Arguments

`DGUIDs`	census year to get the data for, right now only 2021 is supported. Valid DGUIDs for a given geographic level can be queried via ‘get_statcan_wds_metadata()‘.
`members`	list of Member IOs to download data for. By default all characteristics are downloaded. Valid Member IDs and their descriptions can be queried via the ‘get_statcan_wds_metadata()‘ call.
`gender`	optionally query data for only one gender. By default this queries data for all genders, possible values are "Total", "Male", "Female" to only query total data, or for males only or for females only.
`language`	specify language for geography and characteristic names that get added, valid choices are "en" and "fr"
`refresh`	default is ‘FALSE‘ will refresh the temporary cache if ‘TRUE‘

Value

tibble with the enriched census data

Examples

```
# get data for federal electoral district 2013A000459021
## Not run:
get_statcan_wds_data(DGUIDs="2013A000459021",level="FED")

## End(Not run)
```

`get_statcan_wds_metadata`

Query the StatCan WDS for metadata

Description

Get official metadata information from Statistics Canada for a given geographic level. Only available for the 2021 census. Data is cached for the duration of the R session.

Usage

```
get_statcan_wds_metadata(census_year, level, refresh = FALSE)
```

Arguments

`census_year`	census year to get the data for, right now only 2021 is supported
`level`	geographic level to return the data for, valid choices are "PR","CD","CMACA","CSD","CT","ADA","DA"
`refresh`	default is 'FALSE' will refresh the temporary cache if 'TRUE'

Value

tibble with the metadata

Examples

```
# get metadata for federal electoral districts
## Not run:
get_statcan_wds_metadata(census_year="2021",level="FED")

## End(Not run)
```

label_vectors — *Return Census variable names and labels as a tidy data frame*

Description

Return Census variable names and labels as a tidy data frame

Usage

```
label_vectors(x)
```

Arguments

x	A data frame, `sp` or `sf` object returned from `get_census` or similar.

Value

A data frame with a column `variable` containing the truncated variable name, and a column `label` describing it.

Examples

```
## Not run:
# Query census data with truncated labels:
label_data <- get_census(dataset='CA16', regions=list(CMA="59933"),
                         vectors=c("v_CA16_408","v_CA16_409","v_CA16_410"),
                         level='CSD', geo_format = "sf", labels="short")

# Get details for truncated vectors:
label_vectors(label_data)

## End(Not run)
```

list_cancensus_cache — *List cached files*

Description

Lists all cached data and metadata if available

Usage

```
list_cancensus_cache()
```

Value

tibble with metadata on cached data

Examples

```
# list add the cached census data
list_cancensus_cache()
```

`list_census_datasets` *Query the CensusMapper API for available datasets.*

Description

Query the CensusMapper API for available datasets.

Usage

```
list_census_datasets(use_cache = TRUE, quiet = FALSE)
```

Arguments

`use_cache`	If set to TRUE (the default), data will be read from a temporary local cache for the duration of the R session, if available. If set to FALSE, query the API for the data, and refresh the temporary cache with the result.
`quiet`	When TRUE, suppress messages and warnings.

Value

Returns a data frame with a column `dataset` containing the code for the dataset, a column `description` describing it, a `geo_dataset` column identifying the geography dataset the data is based on, a `attribution` column with an attribution string, a `reference` column with a reference identifier, and a `reference_url` column with a link to reference materials.

Examples

```
# List available datasets in CensusMapper
list_census_datasets()
```

list_census_regions *Query the CensusMapper API for available regions in a given dataset.*

Description

Query the CensusMapper API for available regions in a given dataset.

Usage

```
list_census_regions(dataset, use_cache = TRUE, quiet = FALSE)
```

Arguments

dataset	The dataset to query for available regions, e.g. "CA16".
use_cache	If set to TRUE (the default), data will be read from a local cache that is maintained for the duration of the R session, if available. If set to FALSE, query the API for the data, and refresh the local cache with the result.
quiet	When TRUE, suppress messages and warnings.

Value

Returns a data frame with the following columns:

- `region` The region identifier.
- `name` The name of that region.
- `level` The census aggregation level of that region.
- `pop` The population of that region.
- `municipal_status` Additional identifiers to distinguish the municipal status of census subdivisions.
- `CMA_UID` The identifier for the Census Metropolitan Area the region is in (if any).
- `CD_UID` The identifier for the Census District the region is in (if any).
- `PR_UID` The identifier for the Province the region is in (if unique).

Examples

```
list_census_regions('CA16')
```

`list_census_vectors` *Query the CensusMapper API for available vectors for a given dataset.*

Description

Query the CensusMapper API for available vectors for a given dataset.

Usage

```
list_census_vectors(dataset, use_cache = TRUE, quiet = TRUE)
```

Arguments

`dataset`	The dataset to query for available vectors, e.g. `"CA16"`.
`use_cache`	If set to TRUE (the default), data will be read from a local cache that is maintained for the duration of the R session, if available. If set to FALSE, query the API for the data, and refresh the local cache with the result.
`quiet`	When FALSE, shows messages and warnings. Set to TRUE by default.

Value

Returns a data frame detailing the available Census vectors (i.e. variables) for a given Census dataset. This data frame has columns `vector` containing the short code for the variable, `type` describing whether it's a female, male, or total aggregate, `label` indicating the name of the variable, `units` indicating whether the value represents a numeric integer, percentage, dollar figure, or ratio, `parent_vector` to show hierarchical relationship, `aggregation` indicating whether the value is additive or a transformation, and a column `details` with a detailed description of the variable generated by traversing all labels within its hierarchical structure.

Examples

```
## Not run:
# List all vectors for a given Census dataset in CensusMapper
list_census_vectors('CA16')

## End(Not run)
```

`list_recalled_cached_data` *List recalled data stored in local cache*

Description

Checks the local cached database for recalled data and lists all recalled cached entries

Usage

```
list_recalled_cached_data(
  cached_data = list_cancensus_cache(),
  warn_only_once = FALSE
)
```

Arguments

`cached_data`	List of locally cached data to check for recall, default is 'list_cancensus_cache()' in which case it will get checked against all locally cached data
`warn_only_once`	Will only warn on first run during each session, 'FALSE' by default

Value

tibble with rows describing locally cached recalled data

Examples

```
## Not run:
list_recalled_cached_data()

## End(Not run)
```

`parent_census_vectors` *List all parent variables from vector hierarchies given either a list of Census variables returned by* `list_census_vectors`*,* `search_census_vectors`*,* `find_census_vectors`*, or a direct string reference to the vector code.*

Description

List all parent variables from vector hierarchies given either a list of Census variables returned by `list_census_vectors`, `search_census_vectors`, `find_census_vectors`, or a direct string reference to the vector code.

Usage

```
parent_census_vectors(vector_list)
```

Arguments

vector_list — The list of vectors to be used, either a character vector or a filtered tibble as returned from `list_census_vectors`.

Examples

```
# Query parent vectors directly using vector identifier
parent_census_vectors("v_CA16_2519")
## Not run:
# Example using multiple vectors coerced into a list
parent_census_vectors(c("v_CA16_2519","v_CA16_2520","v_CA16_2521"))

# or, equivalently
selected_vectors <- c("v_CA16_2519","v_CA16_2520","v_CA16_2521")
parent_census_vectors(selected_vectors)

# Example using dplyr and piped arguments
library(dplyr, warn.conflicts = FALSE)

list_census_vectors("CA16") %>%
  filter(vector == "v_CA16_2519") %>%
  parent_census_vectors()

## End(Not run)
```

---

remove_from_cancensus_cache

Remove cached files

Description

Remove cached data for paths given

Usage

```
remove_from_cancensus_cache(paths)
```

Arguments

paths — list of paths to remove, as returned by the path column in ‘list_cancensus_cache‘

Value

freed-up disk space

Examples

```
## Not run:
# remove the first cached dataset
cache_data <- list_cancensus_cache()

remove_from_cancensus_cache(cache_data$path[1])

## End(Not run)
```

`remove_recalled_cached_data`
Remove recalled data from local cache

Description

Checks the local cached database for recalled data and removes cached data that has been recalled

Usage

```
remove_recalled_cached_data()
```

Value

Storage size of removed locally cached data that got freed up in number of bytes.

Examples

```
## Not run:
remove_recalled_cached_data()

## End(Not run)
```

`search_census_regions` *Query the CensusMapper API for regions with names matching a searchterm.*

Description

Runs a query against the CensusMapper API to retrieve region data with names matching specific queries. Users can optionally specify the target geography level (e.g. `level = 'CMA'`, `level = 'CSD'`, etc.). Alternatively, calling `explore_census_vectors()` will launch the interactive region selection tool on the CensusMapper site in a new web page or tab.

Usage

```
search_census_regions(searchterm, dataset, level = NA, ...)
```

Arguments

searchterm	The term to search for e.g. "Victoria". Search terms are case insensitive. If unable to find a given search term, this function will suggest the correct spelling to use when possible.
dataset	The dataset to query for available regions, e.g. "CA16".
level	One of NA, 'C', 'PR', 'CMA', 'CD', or 'CSD'. If specified, only return variables of specified ‘level‘.
...	Further arguments passed on to list_census_regions.

Value

A census region list of the same format as ‘list_census_regions()‘ containing the matches.

Examples

```
## Not run:
# This will return a warning that no match was found, but will suggest similar named regions.
search_census_regions('Victorea', 'CA16')

# This will limit region results to only include CMA level regions
search_census_regions('Victoria', 'CA16', level = "CMA")

## End(Not run)
```

---

search_census_vectors   *Query the CensusMapper API for vectors with descriptions matching a search term or phrase (deprecated)*

Description

Query the CensusMapper API for vectors with descriptions matching a search term or phrase (deprecated)

Usage

```
search_census_vectors(searchterm, dataset, type = NA, ...)
```

Arguments

searchterm	The term or phrase to search for e.g. "Ojibway". Search terms are case insensitive. If unable to find a given string, this function will suggest similarly named objects.
dataset	The dataset to query for available vectors, e.g. "CA16".
type	One of NA, 'Total', 'Male' or 'Female'. If specified, only return variables of specified ‘type‘.
...	Further arguments passed on to list_census_vectors.

Examples

```
search_census_vectors('Ojibway', 'CA16')
## Not run:
# This will return a warning that no match was found, but will suggest similar terms.
search_census_vectors('Ojibwe', 'CA16', 'Total')

## End(Not run)
```

`set_cancensus_api_key` *Set Censusmapper API key*

Description

Cancensus requires a free Censusmapper API key to retrieve data. This function helps set the key for either the duration of the session (default) or permanently for use across sessions.

Usage

```
set_cancensus_api_key(key, overwrite = FALSE, install = FALSE)
```

Arguments

`key`	a Censusmapper API key. For more information on keys see the API key section
`overwrite`	Option to overwrite any existing Censusmapper keys already stored locally.
`install`	Option to install permanently for use across sessions.

Examples

```
## Not run:
set_cancensus_api_key("YOUR_CM_API_KEY")

# This will set the key permanently until overwritten again
set_cancensus_api_key("YOUR_CM_API_KEY", install = TRUE)

## End(Not run)
```

`set_cancensus_cache_path` *Set persistent cancensus cache location*

Description

Cancensus provides session caching for retrieved data to increase speeds and reduce API usage. This function will create a persistent cache across sessions.

Usage

```
set_cancensus_cache_path(cache_path, overwrite = FALSE, install = FALSE)
```

Arguments

cache_path	a local directory to use for saving cached data
overwrite	Option to overwrite any existing cache path already stored locally.
install	Option to install permanently for use across sessions.

Examples

```
## Not run:
set_cancensus_cache_path("~/cancensus_cache")

# This will set the cache path permanently until ovewritten again
set_cancensus_cache_path("~/cancensus_cache", install = TRUE)

## End(Not run)
```

---

show_cancensus_api_key

View saved Censusmapper API key

Description

View saved API key'

Usage

```
show_cancensus_api_key()
```

Examples

```
show_cancensus_api_key()
```

`show_cancensus_cache_path` *View saved cache directory path*

Description

View saved cache path'

Usage

```
show_cancensus_cache_path()
```

Examples

```
show_cancensus_cache_path()
```

Index