Package ‘cbsodataR’

February 20, 2020

Type Package

Title Statistics Netherlands (CBS) Open Data API Client

Version 0.4.1

Description The data and meta data from Statistics Netherlands (<https://www.cbs.nl>) can be browsed and downloaded. The client uses the open data API of Statistics Netherlands.

License GPL-2

URL https://github.com/edwindj/cbsodataR

BugReports https://github.com/edwindj/cbsodataR/issues

Encoding UTF-8

Imports whisker, jsonlite, utils

Suggests knitr, rmarkdown, dplyr, shiny, testthat (>= 2.1.0)

VignetteBuilder knitr

RoxygenNote 7.0.2

NeedsCompilation no

Author Edwin de Jonge [aut, cre], Sara Houweling [ctb]

Maintainer Edwin de Jonge <edwindjonge@gmail.com>

Repository CRAN

Date/Publication 2020-02-20 14:50:08 UTC

R topics documented:

cbsodataR-package ......................................................... 2
cache_clear ............................................................... 3
cbs_add_date_column .................................................... 3
cbs_add_label_columns .................................................. 4
cbs_default_selection ................................................... 5
cbs_download_data ....................................................... 6
cbs_download_meta ....................................................... 7

cbsodataR-package

Download all data from Statistics Netherlands / CBS

Description

cbsodataR allows to download all official statistics of Statistics Netherlands (CBS) into R. For an introduction please visit the vignette: vignette("cbsodataR", package="cbsodataR"). The functions `cbs_get_toc()` and `cbs_get_data()` should get you going.

Catalog function

- `cbs_get_toc()`, returns a data.frame with table of contents (toc): the publication meta data for all available tables

Data retrieval

- `cbs_get_data()`, returns the data of a specific opendata/StatLine table
- `cbs_download_table()`, saves the data (and metadata) as csv files into a directory

Meta data

- `cbs_get_meta()`, returns the meta data objects of a specific opendata / StatLine table.
- `cbs_add_date_column()`, converts date/period codes into DateTime objects in the data set that was downloaded.
- `cbs_add_label_columns()`, adds labels to the code columns in the data that was downloaded.
**cache_clear**

clears the cache

---

**Description**

clears the cache

**Usage**

cache_clear()

---

**cbs_add_date_column**  
*Convert the time variable into either a date or numeric.*

---

**Description**

Time periods in data of CBS are coded: yyyyXXww (e.g. 2018J00, 2018M10, 2018W02), which contains year (yyyy), type (XX) and index (ww). `cbs_add_date_column` converts these codes into a `Date()` or numeric. In addition it adds a frequency column denoting the type of the column.

**Usage**

cbs_add_date_column(x, date_type = c("Date", "numeric"), ...)

**Arguments**

- `x`  
data.frame retrieved using `cbs_get_data()`

- `date_type`  
Type of date column: "Date", "numeric". Numeric creates a fractional number which signs the "middle" of the period. e.g. 2018J00 -> 2018.5 and 2018W01 -> 2018.167. This is for the following reasons: otherwise 2018.0 could mean 2018, 2018 Q1 or 2018 Jan, and furthermore 2018.75 is a bit strange for 2018 Q4. If all codes in the dataset have frequency "Y" the numeric output will be integer.

- `...`  
future use.
Value

original dataset with two added columns: <period> date and <period> freq. This last column is a factor with levels: Y, Q and M

See Also

Other data retrieval: `cbs_add_label_columns()`, `cbs_download_data()`, `cbs_extract_table_id()`, `cbs_get_data()`

Other meta data: `cbs_add_label_columns()`, `cbs_download_meta()`, `cbs_get_meta()`

Examples

```r
## Not run:
x <- cbs_get_data( id = "7196ENG" # table id
  , Periods = "2000MM03" # March 2000
  , CPI = "000000" # Category code for total
)

# add a Periods_Date column
x <- cbs_add_date_column(x)
x

# add a Periods_numeric column
x <- cbs_add_date_column(x, date_type = "numeric")
x

## End(Not run)
```

---

cbs_add_label_columns  For each column with codes add label column to data set

Description

Adds cbs labels to the dataset that was retrieved using `cbs_get_data()`.

Usage

cbs_add_label_columns(x, columns = colnames(x), ...)

Arguments

- `x`  data.frame retrieved using `cbs_get_data()`.
- `columns`  character with the names of the columns for which labels will be added
- `...`  not used.
Details

Code columns will be translated into label columns for each of the column that was supplied.
By default all code columns will be accompanied with a label column. The name of each label
column will be `<code_column>_label`.

Value

the original data.frame `x` with extra label columns. (see description)

See Also

Other data retrieval: `cbs_add_date_column()`, `cbs_download_data()`, `cbs_extract_table_id()`,
`cbs_get_data_from_link()`, `cbs_get_data()`
Other meta data: `cbs_add_date_column()`, `cbs_download_meta()`, `cbs_get_meta()`

Examples

```r
## Not run:
# get data for main (000000) Consumer Price Index (7196ENG) for March 2000,
x <- cbs_get_data( id = "7196ENG"
  , Periods = "2000MM03"  # March 2000
  , CPI = "000000"      # main price index
)
cbs_add_label_columns(x)

## End(Not run)
```

---

cbs_default_selection  
*extract the default selection from a cbsodata meta object*

Description

extract the default selection from a cbsodata meta object

Usage

cbs_default_selection(x, ...)

Arguments

x  

meta object

...  

for future use
cbs_download_data  \hspace{5pt} \textit{Gets all data from a cbs table.}

\underline{Description}

Gets all data via bulk download. \texttt{cbs_download_data} dumps the data in (international) csv format.

\underline{Usage}

\begin{verbatim}
cbs_download_data(
  id,
  path = file.path(id, "data.csv"),
  ...,  \hspace{10pt} \textit{optional filter statements to select rows of the data,}
  select = NULL,
  typed = TRUE,
  verbose = FALSE,
  base_url = getOption("cbsodataR.base_url", BASE_URL)
)
\end{verbatim}

\underline{Arguments}

\begin{description}
\item[id] of cbs open data table
\item[path] of data file, defaults to "id/data.csv"
\item[...] optional names of columns to be returned.
\item[select] optional filter statements to select rows of the data,
\item[typed] Should the data automatically be converted into integer and numeric?
\item[verbose] show the underlying downloading of the data
\item[base_url] optionally specify a different server. Useful for third party data services implementing the same protocol.
\end{description}

\underline{See Also}

Other download: \texttt{cbs_download_meta()}, \texttt{cbs_download_table()}

Other data retrieval: \texttt{cbs_add_date_column()}, \texttt{cbs_add_label_columns()}, \texttt{cbs_extract_table_id()}, \texttt{cbs_get_data_from_link()}, \texttt{cbs_get_data()}
cbs_download_meta

Dumps the meta data into a directory

Description

Dumps the meta data into a directory

Usage

```r
cbs_download_meta(
  id,
  dir = id,
  ..., # not used
  verbose = FALSE,
  cache = FALSE,
  base_url = getOption("cbsodataR.base_url", BASE_URL)
)
```

Arguments

- `id`: Id of CBS open data table (see `cbs_get_toc()`)
- `dir`: Directory in which data should be stored. By default it creates a sub directory with the name of the id
- `...`: not used
- `verbose`: Print extra messages what is happening.
- `cache`: Should meta data be cached?
- `base_url`: optionally allow to specify a different server. Useful for third party data services implementing the same protocol.

Value

meta data object

See Also

Other meta data: `cbs_add_date_column()`, `cbs_add_label_columns()`, `cbs_get_meta()`

Other download: `cbs_download_data()`, `cbs_download_table()`
Description

cbs_download_table downloads the data and metadata of a table from statistics Netherlands and stores it in csv format.

Usage

cbs_download_table(
  id,
  ...,  
  dir = id,  
  cache = FALSE, 
  verbose = TRUE,  
  typed = FALSE, 
  base_url = getOption("cbsodataR.base_url", BASE_URL)
)

Arguments

  id  
  Identifier of CBS table (can be retrieved from cbs_get_toc())

  ...  
  Parameters passed on to cbs_download_data()

  dir  
  Directory where table should be downloaded

  cache  
  If metadata is cached use that, otherwise download meta data

  verbose  
  Print extra messages what is happening.

  typed  
  Should the data automatically be converted into integer and numeric?

  base_url  
  optionally specify a different server. Useful for third party data services implementing the same protocol.

Details

cbs_download_table retrieves all raw meta data and data and stores these as csv files in the directory specified by dir. It is possible to add a filter. A filter is specified with `<column_name> = <values>` in which `<values>` is a character vector. Rows with values that are not part of the character vector are not returned.

Value

  meta data object of id cbs_get_meta().

See Also

Other download: cbs_download_data(), cbs_download_meta()
Examples

## Not run:

```r
# download meta data and data from inflation/Consumer Price Indices
download_table(id="7196ENG")
```

## End(Not run)

cbs_extract_table_id

extract the id of a cbs table from the statline url

description

extract the id of a cbs table from the statline url

Usage

cbs_extract_table_id(url, ...)

Arguments

url character with hyperlink to StatLine table

... future use.

Value

character with id, will be NA if not found.

See Also

Other data retrieval: `cbs_add_date_column()`, `cbs_add_label_columns()`, `cbs_download_data()`, `cbs_get_data_from_link()`, `cbs_get_data()`

cbs_get_data

Get data from Statistics Netherlands (CBS)

description

Retrieves data from a table of Statistics Netherlands. A list of available tables can be retrieved with `cbs_get_toc()`. Use the Identifier column of `cbs_get_toc` as id in `cbs_get_data` and `cbs_get_meta`. 
Usage

cbs_get_data(
id,
...,  
select = NULL,
typed = TRUE,
add_column_labels = TRUE,
dir = tempdir(),
verbose = FALSE,
base_url = getOption("cbsodataR.base_url", BASE_URL),
include_ID = FALSE
)

Arguments

id Identifier of table, can be found in `cbs_get_toc()`
...
select optional filter statements, see details.
typed Should the data automatically be converted into integer and numeric?
add_column_labels Should column titles be added as a label (TRUE) which are visible in `View`? 
dir Directory where the table should be downloaded. Defaults to temporary directory 
verbose Print extra messages what is happening.
base_url optionally specify a different server. Useful for third party data services implementing the same protocol.
include_ID Should the data include the ID column for the rows?

Details

To reduce the download time, optionally the data can be filtered on category values: for large tables (> 100k records) this is a wise thing to do.

The filter is specified with (see examples below):

- `<column_name> = <values>` in which `<values>` is a character vector. Rows with values that are not part of the character vector are not returned. **Note that the values have to be values from the $Key column of the corresponding meta data. These may contain trailing spaces...**

- `<column_name> = has_substring(x)` in which `x` is a character vector. Rows with values that do not have a substring that is in `x` are not returned. Useful substrings are "JJ", "KW", "MM" for Periods (years, quarters, months) and "PV", "CR" and "GM" for Regions (provinces, corops, municipalities).

- `<column_name> = eq(<values>) | has_substring(x)`, which combines the two statements above.

By default the columns will be converted to their type (`typed=TRUE`). CBS uses multiple types of missing (unknown, surpressed, not measured, missing): users wanting all these nuances can use `typed=FALSE` which results in character columns.
Value

data.frame with the requested data. Note that a csv copy of the data is stored in dir.

Note

All data are downloaded using cbs_download_table()

See Also

cbs_get_meta(), cbs_download_data()

Other data retrieval: cbs_add_date_column(), cbs_add_label_columns(), cbs_download_data(), cbs_extract_table_id(), cbs_get_data_from_link()

Other query: eq(), has_substring()

Examples

## Not run:
cbs_get_data(id = "7196ENG" # table id
  , Periods = "2000MM03" # March 2000
  , CPI = "000000" # Category code for total
)

# useful substrings:
## Periods: "JJ": years, "KW": quarters, "MM", months
## Regions: "NL", "PV": provinces, "GM": municipalities

cbs_get_data(id = "7196ENG" # table id
  , Periods = has_substring("JJ") # all years
  , CPI = "000000" # Category code for total
)

cbs_get_data(id = "7196ENG" # table id
  , Periods = c("2000MM03","2001MM12") # March 2000 and Dec 2001
  , CPI = "000000" # Category code for total
)

cbs_get_data(id = "7196ENG" # table id
  , Periods = eq("2000MM01") | has_substring("JJ") # Jan 2000 and all years
  , CPI = "000000" # Category code for total
)

## End(Not run)
cbs_get_meta

Description

Retrieve the meta data of a CBS open data table. Caching (cache=TRUE) improves the performance considerably.

Usage

```r
cbs_get_meta(
  id,
  verbose = FALSE,
  cache = TRUE,
  base_url = getOption("cbsodataR.base_url", BASE_URL)
)
```

See Also

Other data retrieval: `cbs_add_date_column()`, `cbs_add_label_columns()`, `cbs_download_data()`, `cbs_extract_table_id()`, `cbs_get_data()`
Arguments

- **id**: internal id of CBS table, can be retrieved with `cbs_get_toc()`
- **verbose**: Print extra messages what is happening.
- **cache**: should the result be cached?
- **base_url**: optionally specify a different server. Useful for third party data services implementing the same protocol.

Details

The meta data of a CBS table is determined by the web api of Statistics Netherlands. `cbsodataR` stays close to this API. Each `cbsodataR` object has the following metadata items, which are all `data.frame`

- **$TableInfos**: data.frame with the descriptive publication metadata of the table, such as Title, Description, Summary etc.
- **$DataProperties**: data.frame with the Title, Description, Unit etc. of each column in the dataset that is downloaded with `cbs_get_data()`.
- **$CategoryGroups**: hierarchical groupings of the code columns.
- **$<code column>**: for each code column a data.frame with the Title, Key, Description etc. of each code / category in that column. E.g. Perioden for time codes `c("2019JJ00","2018JJ00")`.

Value

cbs_table object containing several data.frames with meta data (see details)

See Also

Other meta data: `cbs_add_date_column()`, `cbs_add_label_columns()`, `cbs_download_meta()`

---

**cbs_get_meta_from_dir**  
*Load meta data from a downloaded table*

Description

Load meta data from a downloaded table

Usage

`cbs_get_meta_from_dir(dir)`

Arguments

- **dir**: Directory where data was downloaded

Value

cbs_table object with meta data
cbs_get_tables_themes  Get a the list of tables connected to themes

Description
Get a the list of tables connected to themes

Usage
cbs_get_tables_themes(
  ...,  
  select = NULL,  
  verbose = FALSE,  
  cache = TRUE,  
  base_url = getOption("cbsodataR.base_url", BASE_URL)
)

Arguments
... Use this to add a filter to the query e.g. get_tables_themes(ID=10).
select character vector with names of wanted properties. default is all
verbose Print extra messages what is happening.
cache Should the result be cached?
base_url optionally specify a different server. Useful for third party data services implement-ing the same protocal.

Value
A data.frame with various properties of SN/CBS themes.

cbs_get_themes  Get list of all cbs thematic entries.

Description
Returns a list of all cbs themes.

Usage
cbs_get_themes(
  ...,  
  select = NULL,  
  verbose = TRUE,  
  cache = FALSE,  
  base_url = getOption("cbsodataR.base_url", BASE_URL)
)
cbs_get_toc

Arguments

... Use this to add a filter to the query e.g. get_themes(ID=10).
select character vector with names of wanted properties. default is all
verbose Print extra messages what is happening.
   cache Should the result be cached?
   base_url optionally specify a different server. Useful for third party data services implement-
download_data-deprecated

Arguments

... filter statement to select rows, e.g. Language="nl"
convert_dates convert the columns with date-time information into DateTime (default TRUE)
select character columns to be returned, by default all columns will be returned.
verbose logical prints the calls to the webservice
cache logical should the result be cached?
base_url optionally specify a different server. Useful for third party data services implementing the same protocol.
include_ID logical column needed by OData but with no current use.

Value
data.frame with identifiers, titles and descriptions of tables

Note

cbs_get_toc will cache results, so subsequent calls will be much faster.

Examples

## Not run:

# get list of english tables
tables_en <- cbs_get_toc(Language="en")

# get list of dutch tables
tables_nl <- cbs_get_toc(Language="nl")
View(tables_nl)

## End(Not run)

download_data-deprecated

*Gets all data from a cbs table.*

Description

This method is deprecated in favor of `cbs_download_data()`.

Usage

download_data(
id,
path = file.path(id, "data.csv"),
..., 
select = NULL,
download_meta-deprecated

Dumps the meta data into a directory

Description

This method is deprecated in favor of cbs_download_meta().

Usage

download_meta(
  id,
  dir = id,
  ...,  
  verbose = FALSE,
  cache = FALSE,
  base_url = getOption("cbsodataR.base_url", BASE_URL)
)

Arguments

id of cbs open data table
path of data file, defaults to "id/data.csv"
... optional filter statements to select rows of the data,
select optional names of columns to be returned.
typed Should the data automatically be converted into integer and numeric?
verbose show the underlying downloading of the data
base_url optionally specify a different server. Useful for third party data services implementing the same protocol.

See Also

Other download: cbs_download_meta(), cbs_download_table()
Other data retrieval: cbs_add_date_column(), cbs_add_label_columns(), cbs_extract_table_id(), cbs_get_data_from_link(), cbs_get_data()
Arguments

- **id**: Id of CBS open data table (see `cbs_get_toc()`)
- **dir**: Directory in which data should be stored. By default it creates a sub directory with the name of the id
- **...**: not used
- **verbose**: Print extra messages what is happening.
- **cache**: Should meta data be cached?
- **base_url**: optionally allow to specify a different server. Useful for third party data services implementing the same protocol.

Value

- meta data object

See Also

- Other meta data: `cbs_add_date_column()`, `cbs_add_label_columns()`, `cbs_get_meta()`
- Other download: `cbs_download_data()`, `cbs_download_table()`

Description

This method is deprecated in favor of `cbs_download_table()`.

Usage

```r
download_table(
  id,
  ..., 
  dir = id,
  cache = FALSE,
  verbose = TRUE,
  typed = FALSE,
  base_url = getOption("cbsodataR.base_url", BASE_URL)
)
```
eq

Arguments

- **id**: Identifier of CBS table (can be retrieved from `cbs_get_toc()`)
- **...**: Parameters passed on to `cbs_download_data()`
- **dir**: Directory where table should be downloaded
- **cache**: If metadata is cached use that, otherwise download metadata
- **verbose**: Print extra messages what is happening.
- **typed**: Should the data automatically be converted into integer and numeric?
- **base_url**: Optionally specify a different server. Useful for third party data services implementing the same protocol.

Details

cbs_download_table retrieves all raw meta data and data and stores these as csv files in the directory specified by `dir`. It is possible to add a filter. A filter is specified with `<column_name> = <values>` in which `<values>` is a character vector. Rows with values that are not part of the character vector are not returned.

Value

- meta data object of id `cbs_get_meta()`.

See Also

Other download: `cbs_download_data()`, `cbs_download_meta()`

Examples

```r
## Not run:
# download meta data and data from inflation/Consumer Price Indices
download_table(id="7196ENG")
## End(Not run)
```

---

**eq**

*Detect codes in a column*

Description

Detects for codes in a column. `eq` filters the data set at CBS: rows that have a code that is not in `x` are filtered out.

Usage

```r
eq(x, column = NULL)
```
get_data-deprecated

Get data from Statistics Netherlands (CBS)

Description

This method is deprecated in favor of cbs_get_data()
Usage

get_data(
  id,
  ..., 
  recode = TRUE,
  use_column_title = recode,
  dir = tempdir(),
  base_url = getOption("cbsodataR.base_url", BASE_URL)
)

Arguments

id  Identifier of table, can be found in cbs_get_toc()
...
optional filter statements, see details.
recode recodes all codes in the code columns with their Title as found in the metadata
use_column_title
  not used.
dir  Directory where the table should be downloaded. Defaults to temporary directory
base_url optionally specify a different server. Useful for third party data services implementing the same protocol.

Details

To reduce the download time, optionally the data can be filtered on category values: for large tables (> 100k records) this is a wise thing to do.

The filter is specified with (see examples below):

- `<column_name> = <values>` in which `<values>` is a character vector. Rows with values that are not part of the character vector are not returned. **Note that the values have to be values from the $Key column of the corresponding metadata. These may contain trailing spaces...**
- `<column_name> = has_substring(x) in which x is a character vector. Rows with values that do not have a substring that is in x are not returned. Useful substrings are "JJ", "KW", "MM" for Periods (years, quarters, months) and "PV", "CR" and "GM" for Regions (provinces, corops, municipalities).
- `<column_name> = eq(<values>) | has_substring(x), which combines the two statements above.

By default the columns will be converted to their type (typed=TRUE). CBS uses multiple types of missing (unknown, suppressed, not measured, missing): users wanting all these nuances can use typed=FALSE which results in character columns.

Value

data.frame with the requested data. Note that a csv copy of the data is stored in dir.
See Also

cbs_get_meta(), cbs_download_data()

Other data retrieval: cbs_add_date_column(), cbs_add_label_columns(), cbs_download_data(),
cbs_extract_table_id(), cbs_get_data_from_link()

Other query: eq(), has_substring()

Examples

```r
## Not run:
cbs_get_data( id = "7196ENG" # table id
  , Periods = "2000MM03" # March 2000
  , CPI = "000000" # Category code for total
)

# useful substrings:
## Periods: "JJ": years, "KW": quarters, "MM", months
## Regions: "NL", "PV": provinces, "GM": municipalities

cbs_get_data( id = "7196ENG" # table id
  , Periods = has_substring("JJ") # all years
  , CPI = "000000" # Category code for total
)

cbs_get_data( id = "7196ENG" # table id
  , Periods = c("2000MM03","2001MM12") # March 2000 and Dec 2001
  , CPI = "000000" # Category code for total
)

cbs_get_data( id = "7196ENG" # table id
  , Periods = eq("2000MM01") | has_substring("JJ") # Jan 2000 and all years
  , CPI = "000000" # Category code for total
)

## End(Not run)
```

get_meta-deprecated  Get meta data from table

Description

This method is deprecated in favor of cbs_get_meta()

Usage

```r
get_meta(
  id,
  verbose = TRUE,
  cache = FALSE,
  base_url = getOption("cbsdataR.base_url", BASE_URL)
)
```
**get_meta_from_dir**

**Arguments**

- **id**: internal id of CBS table, can be retrieved with `cbs_get_toc()`
- **verbose**: Print extra messages what is happening.
- **cache**: should the result be cached?
- **base_url**: optionally specify a different server. Useful for third party data services implementing the same protocol.

**Details**

The meta data of a CBS table is determined by the web api of Statistics Netherlands. `cbsodataR` stays close to this API. Each `cbsodataR` object has the following metadata items, which are all `data.frame`

- **$TableInfos**: data.frame with the descriptive publication metadata of the table, such as `Title`, `Description`, `Summary` etc.
- **$DataProperties**: data.frame with the `Title`, `Description`, `Unit` etc. of each column in the dataset that is downloaded with `cbs_get_data()`.
- **$CategoryGroups**: hierarchical groupings of the code columns.
- **$<code column>**: for each code column a data.frame with the `Title`, `Key`, `Description` etc. of each code / category in that column. e.g. `Perioden` for time codes `c("2019J00","2018J00")`.

**Value**

cbs_table object containing several data.frames with meta data (see details)

**See Also**

Other meta data: `cbs_add_date_column()`, `cbs_add_label_columns()`, `cbs_download_meta()`

---

**Description**

Load meta data from a downloaded table

**Usage**

get_meta_from_dir(dir)

**Arguments**

- **dir**: Directory where data was downloaded

**Value**

cbs_table object with meta data
get_tables_themes  Get a the list of tables connected to themes

Description
Get a the list of tables connected to themes

Usage
get_tables_themes(
  ..., 
  select = NULL, 
  base_url = getOption("cbsodataR.base_url", BASE_URL) 
)

Arguments
...
Use this to add a filter to the query e.g. get_tables_themes(ID=10).
select character vector with names of wanted properties. default is all
base_url optionally specify a different server. Useful for third party data services implementing the same protocol.

Value
A data.frame with various properties of SN/CBS themes.

get_table_list  Retrieve a data.frame with requested cbs tables

Description
This method is deprecated in favor of cbs_get_toc().

Usage
get_table_list(
  ..., 
  select = NULL, 
  base_url = getOption("cbsodataR.base_url", BASE_URL) 
)

Arguments
...
filter statement to select rows, e.g. Language="nl"
select character columns to be returned, by default all columns will be returned.
base_url optionally specify a different server. Useful for third party data services implementing the same protocol.
get_themes

Value
data.frame with identifiers, titles and descriptions of tables

Examples

## Not run:

# get list of english tables
tables_en <- get_table_list(Language="en")

# get list of dutch tables
tables_nl <- get_table_list(Language="nl")
View(tables_nl)

## End(Not run)

got_themes

Get list of all cbs thematic entries.

Description

Returns a list of all cbs themes.

Usage
got_themes(
  ..., 
  select = NULL, 
  verbose = TRUE, 
  cache = FALSE, 
  base_url = getOption("cbsodataR.base_url", BASE_URL)
)

Arguments

Use this to add a filter to the query e.g. get_themes(ID=10).

select character vector with names of wanted properties. default is all
verbose Print extra messages what is happening.
cache Should the result be cached?
base_url optionally specify a different server. Useful for third party data services implementing the same protocol.

Value

A data.frame with various properties of SN/CBS themes.

The filter is specified with <column_name> = <values> in which <values> is a character vector. Rows with values that are not part of the character vector are not returned.
Examples

```r
## Not run:
# get list of all themes
get_themes()

# get list of all dutch themes from the Catalog "CBS"
get_themes(Language="nl", Catalog="CBS")

## End(Not run)
```

### has_substring

**Detect substring in column**

**Description**

Detects a substring in a column. `has_substring` filters the dataset at CBS: rows that have a code that does not contain (one of) `x` are filtered out.

**Usage**

```r
has_substring(x, column = NULL)
```

**Arguments**

- `x`: substring to be detected in column
- `column`: column name

**See Also**

Other query: `cbs_get_data()`, `eq()`

**Examples**

```r
## Not run:
cbs_get_data( id = "7196ENG" # table id
    , Periods = "2000MM03" # March 2000
    , CPI = "000000" # Category code for total
)

# useful substrings:
## Periods: "JJ": years, "KW": quarters, "MM", months
## Regions: "NL", "PV": provinces, "GM": municipalities
cbs_get_data( id = "7196ENG" # table id
    , Periods = has_substring("JJ") # all years
    , CPI = "000000" # Category code for total
)
cbs_get_data( id = "7196ENG" # table id
```
resolve_deeplink

resolve a deeplink created in the opendata portal

Description

resolve a deeplink created in the opendata portal

Usage

```r
resolve_deeplink(
  deeplink,
  ..., 
  base_url = getOption("cbsodataR.base_url", BASE_URL)
)
```

Arguments

- `deeplink`: url to the deeplink in the opendataportal
- `...`: used in the query
- `base_url`: optionally specify a different server. Useful for third party data services implementing the same protocol.

Value

information object with table id, select, filter and query statement.
Index

cache_clear, 3

cbs_add_date_column, 3, 5–7, 9, 11–13, 17, 18, 22, 23
cbs_add_date_column(), 2
cbs_add_date_columns(), 4, 4, 6, 7, 9, 11–13, 17, 18, 22, 23
cbs_add_label_columns, 2

cbs_add_label_columns(), 2
cbs_default_selection, 5
cbs_download_data, 4, 5, 6, 7–9, 11, 12, 18, 19, 22
cbs_download_data(), 8, 11, 16, 19, 22
cbs_download_data_from_link, 4–6, 9, 11, 12, 17, 18, 19

cbs_download_data_from_link(), 4–6, 9, 11, 12, 17, 18

cbs_download_data_to_html, 4, 5, 6, 7–9, 11, 12, 18, 19

cbs_download_data_to_html(), 8, 11, 16, 19, 22

cbs_download_data_to_pdf, 4–6, 7, 8, 13, 17, 21, 19, 23

cbs_download_data_to_pdf(), 6, 7, 8, 13, 17

cbs_download_table, 6, 7, 8, 17, 18
cbs_download_table(), 2, 11, 18

cbs_extract_table_id, 4–6, 9, 11, 12, 17, 22
cbs_get_data, 4–6, 9, 9, 12, 17, 20, 26
cbs_get_data(), 2–4, 13, 20, 23

cbs_get_data_from_link, 4–6, 9, 11, 17, 22
cbs_get_meta, 4, 5, 7, 12, 18
cbs_get_meta(), 2, 8, 11, 19, 22

cbs_get_meta_from_dir, 13
cbs_get_meta_from_dir(), 13

cbs_get_themes, 24
cbs_get_themes(), 14

cbs_get_toc, 15
cbs_get_toc(), 2, 7–10, 13, 18, 19, 21, 23, 24

cbsodataR (cbsodataR-package), 2
cbsodataR-package, 2

cbsodataR-package, 2

cbsodataR-package, 2

Date(), 3
download_data

download_data (download_data-deprecated), 16
download_data-deprecated, 16
download_data

download_data (download_data-deprecated), 16
download_data

download_data (download_data-deprecated), 16
download_data

download_meta

download_meta (download_meta-deprecated), 17
download_meta

download_meta (download_meta-deprecated), 17
download_meta

download_table

download_table (download_table-deprecated), 18
download_table

download_table (download_table-deprecated), 18
download_table

eq, 11, 19, 22, 26
get_data (get_data-deprecated), 20
get_data

get_data (get_data-deprecated), 20
get_data

get_meta (get_meta-deprecated), 22
get_meta

get_meta (get_meta-deprecated), 22
get_meta

get_meta_from_dir, 23
get_table_list, 24
get_tables_themes, 24
get_themes, 25

has_substring, 11, 20, 22, 26

resolve_deeplink, 27

28