Package: cbsodata4 (via r-universe)

June 23, 2024

Type Package

Title Statistics Netherlands (CBS) Open Data API Client v4

Version 0.9.3.9000

Description The data and meta data from Statistics Netherlands (<https://www.cbs.nl>) can be browsed and downloaded. The client uses the OData v4 API of Statistics Netherlands.

License GPL-3

Encoding UTF-8

LazyData true

Imports jsonlite, data.table

Suggests knitr, rmarkdown, tinytest, covr

URL https://github.com/statistiekcbs/cbsodata4

BugReports https://github.com/statistiekcbs/cbsodata4/issues

RoxygenNote 7.1.2

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

Repository https://edwindj.r-universe.dev

RemoteUrl https://github.com/statistiekcbs/cbsodata4

RemoteRef HEAD

RemoteSha e064dfa5f3cdc33d5040813d53917d23c6c750c5

Contents

`cbs4_add_date_column` *Convert the time variable into either a date or numeric.*

Description

Add extra date columns to data set, for the creation of time series or graphics.

Usage

```
cbs4_add_date_column(data, date_type = c("Date", "numeric"), ...)
```

Arguments

`data`	`data.frame` retrieved using `cbs4_get_data()`
`date_type`	Type of date column: "Date", "numeric". See details.
`...`	future use.

Details

Time periods in data of CBS are coded: yyyyXXww (e.g. 2018JJ00, 2018MM10, 2018KW02), which contains year (yyyy), type (XX) and index (ww). `cbs4_add_date_column` converts these codes into a `Date()` or `numeric`.

`"Date"` will create a date that signifies the start of the period:

- "2018JJ00" will turn into "2018-01-01"
- "2018KW02" will turn into "2018-04-01"

`"numeric"` creates a fractional number which signs the "middle" of the period. e.g. 2018JJ00 -> 2018.5 and 2018KW01 -> 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`.

The `<period_freq>` column indicates the period type / frequency:

- `Y`: year
- `Q`: quarter
- `M`: month
- `W`: week
- `D`: day

Value

original dataset with two added columns: <period>_Date and <period>_freq. See details.

See Also

cbs4_get_metadata()

Other add metadata columns: cbs4_add_label_columns(), cbs4_add_unit_column()

Examples

```
if (interactive()){

  # works on observations...
  obs <- cbs4_get_observations( id        = "80784ned"    # table id
                             , Perioden  = "2019JJ00" # Year 2019
                             , Geslacht  = "1100"       # code for total gender
                             , RegioS    = "NL01"       # code for region NL
                             , Measure   = "M003371_2"
                             )

  # add a Periods_Date column
  obs_d <- cbs4_add_date_column(obs)
  obs_d

  # add a Periods_numeric column
  obs_d <- cbs4_add_date_column(obs, date_type = "numeric")
  obs_d

  # works on data
  d <- cbs4_get_data( id        = "80784ned"    # table id
                    , Perioden  = "2019JJ00"   # Year 2019
                    , Geslacht  = "1100"       # code for total gender
                    , RegioS    = "NL01"       # code for region NL
                    , Measure   = "M003371_2"
                    )
  cbs4_add_date_column(d)
}
```

---

cbs4_add_label_columns

Add understandable labels to a table

Description

Add columns with labels to the dataset.

Usage

```
cbs4_add_label_columns(data, ...)
```

Arguments

data	downloaded with cbs4_get_data()
...	not used

Details

cbs4_add_label_columns() adds for the Measure and each <Dimension> column an extra column MeasureLabel (<Dimension>Label) that contains the Title of each code, making the table more digestible. Title and other metadata can also be found using cbs4_get_metadata().

Value

original dataset with extra label columns. See details.

See Also

cbs4_get_metadata()

Other add metadata columns: cbs4_add_date_column(), cbs4_add_unit_column()

Examples

```
if (interactive()){
  # works on observations
  obs <- cbs4_get_observations("84287NED", Perioden="2019MM12")
  obs # without label columns

  obs_labeled <- cbs4_add_label_columns(obs)
  obs_labeled

  # works on data
  d <- cbs4_get_data("84287NED", Perioden="2019MM12")
  d # cbs4_get_data automagically labels measure columns.

  d_labeled <- cbs4_add_label_columns(d)
  d_labeled
}
```

---

cbs4_add_unit_column *Add unit column to observations*

Description

Add a unit column the unit of each measure.

Usage

```
cbs4_add_unit_column(data, ...)
```

Arguments

data	downloaded with cbs4_get_observations()
...	not used

Details

cbs4_add_unit_column() retrieves the Units for each Measure from MeasureCodes in the metadata (cbs4_get_metadata()) and adds this to the observations data set.

Value

original observations data.frame() with extra Unit column.

See Also

cbs4_get_metadata()

Other add metadata columns: cbs4_add_date_column(), cbs4_add_label_columns()

Examples

```
if (interactive()){
  # works only on observations
  obs <- cbs4_get_observations("84287NED", Perioden="2019MM12")
  obs # without Unit column

  obs_unit <- cbs4_add_unit_column(obs)
  obs_unit # with unit column
}
```

---

cbs4_download *Download observations and metadata*

Description

Download observations and metadata to a directory. This function is the working horse for cbs4_get_data() and cbs4_get_observations() and has many of the same options. This function is useful if you do not want to load an entire dataset into memory, but just download the data and metadata in csv format.

Usage

```
cbs4_download(
  id,
  download_dir = id,
  ...,
  query = NULL,
  catalog = "CBS",
```

```
  show_progress = interactive() && !verbose,
  sep = ",",
  verbose = getOption("cbsodata4.verbose", FALSE),
  base_url = getOption("cbsodata4.base_url", BASEURL4)
)
```

Arguments

id	Identifier of publication
download_dir	directory where files are to be stored
...	optional selection statement to retrieve a subset of the data.
query	optional odata4 query in odata syntax (overwrites any specification in ...)
catalog	Catalog to download from
show_progress	logical if TRUE downloading shows a progress bar. Cannot be used together with verbose=TRUE
sep	seperator to be used in writing the data
verbose	Should messages be printed...
base_url	Possible other website which implements same protocol.

Value

metadata of table (invisible()).

See Also

Other data-download: cbs4_get_data(), cbs4_get_observations()

---

cbs4_get_catalogs *Retrieve all (alternative) catalogs of Statistics Netherlands*

Description

Retrieve catalogs of Statistics Netherlands. Beside the main "CBS" catalog other catalogs contain extra datasets that are not part of the main production of CBS / Statistics Netherlands.

Usage

```
cbs4_get_catalogs(
  base_url = getOption("cbsodata4.base_url", BASEURL4),
  verbose = getOption("cbsodata4.verbose", FALSE)
)
```

Arguments

base_url	possible other url that implements same interface
verbose	if TRUE the communication to the server is shown.

Value

data.frame() with the different catalogs available.

See Also

Other datasets: cbs4_get_datasets()

---

cbs4_get_data *Get data from CBS*

Description

Get data from table id. The data of a CBS opendata table is in so-called wide format. Each Measure has its own column.

Usage

```
cbs4_get_data(
  id,
  catalog = "CBS",
  ...,
  query = NULL,
  name_measure_columns = TRUE,
  show_progress = interactive() && !verbose,
  download_dir = file.path(tempdir(), id),
  verbose = getOption("cbsodata4.verbose", FALSE),
  sep = ",",
  as.data.table = FALSE,
  base_url = getOption("cbsodata4.base_url", BASEURL4)
)
```

Arguments

id	Identifier of the Opendata table. Can be retrieved with cbs4_get_datasets()
catalog	Catalog in which the dataset is to be found.
...	optional selections on data, passed through to cbs4_download. See examples
query	optional query in odata4 syntax (overwrites any specification in ...)
name_measure_columns	logical if TRUE the Title of the measure will be set as name column.
show_progress	if TRUE shows progress of data download, can't be used together with verbose.
download_dir	directory in which the data and metadata is downloaded. By default this is temporary directory, but can be set manually
verbose	if TRUE prints the steps taken to retrieve the data.
sep	separator to be used to download the data.
as.data.table	logical, should the result be of type data.table?
base_url	Possible other url which implements same protocol.

Details

The returned data.frame() has the following columns:

- For each dimension a separate column with category identifiers. Category labels can be added with cbs4_add_label_columns() or found in cbs4_get_metadata(). Date columns can be added with cbs4_add_date_column().
- For each Measure / Topic a separate column with values. Units can be found in cbs4_get_metadata() (MeasureCodes).

For a long format instead of wide format see cbs4_get_observations() which has one Measure column and a Value column.

Value

a data.frame() or data.table() object. See details.

See Also

cbs4_get_metadata()

Other data-download: cbs4_download(), cbs4_get_observations()

Examples

```
if (interactive()){

  # filter on Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = "2019MM12" # december 2019
                )

  # filter on multiple Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = c("2019MM12", "2020MM01") # december 2019, january 2020
                )

  # to filter on a dimension just add the filter to the query

  # filter on Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = "2019MM12" # december 2019
                , BedrijfstakkenBranchesSBI2008 = "T001081"
                )


  # filter on Perioden with contains
  cbs4_get_data("84287NED"
                 , Perioden = contains("2020")
                 , BedrijfstakkenBranchesSBI2008 = "T001081"
  )

  # filter on Perioden with multiple contains
```

```
cbs4_get_data("84287NED"
              , Perioden = contains(c("2019MM1", "2020"))
              , BedrijfstakkenBranchesSBI2008 = "T001081"
)

# filter on Perioden with contains or = "2019MM12
cbs4_get_data("84287NED"
              , Perioden = contains("2020") | "2019MM12"
              , BedrijfstakkenBranchesSBI2008 = "T001081"
)

# This all works on observations too
cbs4_get_observations( id        = "80784ned"     # table id
                     , Perioden  = "2019JJ00"     # Year 2019
                     , Geslacht  = "1100"         # code for total gender
                     , RegioS    = contains("PV") # provinces
                     , Measure   = "M003371_2"    # topic selection
                     )

# supply your own odata 4 query
cbs4_get_data("84287NED", query = "$filter=Perioden eq '2019MM12'")

# an odata 4 query will overrule other filter statements
cbs4_get_data("84287NED"
             , Perioden = "2018MM12"
             , query = "$filter=Perioden eq '2019MM12'"
             )

# With query argument an odata4 expression with other (filter) functions can be used
cbs4_get_observations(
  id     = "80784ned"    # table id
  ,query = paste0(       # odata4 query
     "$skip=4",          # skip the first 4 rows of the filtered result
     "&$top=20",         # then slice the first 20 rows of the filtered result
   "&$select=Measure,Geslacht,Perioden,RegioS,Value", # omit the Id and ValueAttribute fields
     "&$filter=endswith(Measure,'_1')") # filter only Measure ending on '_1'
  )

}
```

`cbs4_get_datasets` *Get available datasets*

Description

Get the available datasets from open data portal statline.

Usage

```
cbs4_get_datasets(
```

```
  catalog = "CBS",
  convert_dates = TRUE,
  verbose = getOption("cbsodata4.verbose", FALSE),
  base_url = getOption("cbsodata4.base_url", BASEURL4)
)

cbs4_get_toc(
  catalog = "CBS",
  convert_dates = TRUE,
  verbose = getOption("cbsodata4.verbose", FALSE),
  base_url = getOption("cbsodata4.base_url", BASEURL4)
)
```

Arguments

catalog	only show the datasets from that catalog. If NULL all datasets of all catalogs will be returned.
convert_dates	Converts date columns in Date-Time type (in stead of character)
verbose	Should the url request be printed?
base_url	base url of the CBS OData 4 API

Details

Setting the catalog to NULL will return all

Value

data.frame() with publication metadata of tables.

Note

the datasets are downloaded only once per R session and cached. Subsequent calls to cbs4_get_datasets will use the results of the first call.

See Also

Other datasets: cbs4_get_catalogs()

Examples

```
if (interactive()){
  # retrieve the main datasets (catalog = "CBS")
  ds <- cbs4_get_datasets()
  print(nrow(ds))

  # see cbs4_get_catalogs() to retrieve all catalogs
  ds_asd <- cbs4_get_datasets(catalog = "CBS-asd")
  print(nrow(ds_asd))

  ds_all <- cbs4_get_datasets(catalog = NULL)
```

```
  print(nrow(ds_all))
}
```

`cbs4_get_metadata` *Retrieve the metadata of a publication*

Description

Retrieve the metadata of a publication. The meta object contains all metadata properties of cbsodata in the form of data.frames.

Usage

```
cbs4_get_metadata(
  id,
  catalog = "CBS",
  ...,
  base_url = getOption("cbsodata4.base_url", BASEURL4),
  verbose = getOption("cbsodata4.verbose", FALSE)
)
```

Arguments

`id`	Identifier of publication or data retrieved with `cbs4_get_data()`/`cbs4_get_observations()`
`catalog`	Catalog, from the set of `cbs4_get_catalogs()`
`...`	not used
`base_url`	alternative url that implements same interface as statistics netherlands.
`verbose`	Should the function report on retrieving the data

Details

Each `data.frame` describes properties of the CBS / Statistics Netherlands table: "Dimensions", "MeasureCodes" and one ore more "\<Dimension\>Codes" describing the meta data of the borders of a CBS table.

Examples

```
if (interactive()){
  meta <- cbs4_get_metadata("80416ned")
  print(names(meta))

  # Dimension columns in the dataset
  meta$Dimensions

  # the metadata of the Measures/Topics
  meta$MeasureCodes
```

```
  # the metadata of the Perioden Categories
  meta$PeriodenCodes

  # all descriptive and publication meta data on this dataset
  meta$Properties
}
```

`cbs4_get_observations` *Get observations from a table.*

Description

Get observations from table `id`. Observations are data of a CBS opendata table in so-called long format.

Usage

```
cbs4_get_observations(
  id,
  ...,
  query = NULL,
  catalog = "CBS",
  download_dir = file.path(tempdir(), id),
  show_progress = interactive() && !verbose,
  verbose = getOption("cbsodata4.verbose", FALSE),
  sep = ",",
  includeId = TRUE,
  as.data.table = FALSE,
  base_url = getOption("cbsodata4.base_url", BASEURL4)
)
```

Arguments

`id`	Identifier of the Opendata table. Can be retrieved with `cbs4_get_datasets()`
`...`	optional selections on data, passed through to cbs4_download. See examples
`query`	optional query in odata4 syntax (overwrites any specification in `...`)
`catalog`	Catalog in which the dataset is to be found.
`download_dir`	directory in which the data and metadata is downloaded. By default this is temporary directory, but can be set manually
`show_progress`	if `TRUE` shows progress of data download, can't be used together with verbose.
`verbose`	if `TRUE` prints the steps taken to retrieve the data.
`sep`	separator to be used to download the data.
`includeId`	`logical`, should the Id column be downloaded?
`as.data.table`	`logical`, should the result be of type data.table?
`base_url`	Possible other url which implements same protocol.

Details

The returned `data.frame()` has the following columns:

- A `Measure` column with identifiers/codes of measures/topics. Detailed information on Measures can be found with in `MeasureCodes` using `cbs4_get_metadata()`. Measure labels can be added with `cbs4_add_label_columns()`.
- A `Value` column with the (numerical) value, Units can be added with `cbs4_add_unit_column()`.
- An optional `ValueAttribute` column with data point specific metadata.
- For each dimension a separate column with category identifiers. Category labels can be added with `cbs4_add_label_columns()` or found in `cbs4_get_metadata()`. Date columns can be added with `cbs4_add_date_column()`.

`cbs4_get_data()` offers an alternative in which each variable/topic/Measure has its own column.

Value

`data.frame()` or `data.table()` object, see details.

See Also

`cbs4_get_metadata()`

Other data-download: `cbs4_download()`, `cbs4_get_data()`

Examples

```
if (interactive()){

  # filter on Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = "2019MM12" # december 2019
                )

  # filter on multiple Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = c("2019MM12", "2020MM01") # december 2019, january 2020
                )

  # to filter on a dimension just add the filter to the query

  # filter on Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = "2019MM12" # december 2019
                , BedrijfstakkenBranchesSBI2008 = "T001081"
                )


  # filter on Perioden with contains
  cbs4_get_data("84287NED"
                 , Perioden = contains("2020")
                 , BedrijfstakkenBranchesSBI2008 = "T001081"
```

```
)

# filter on Perioden with multiple contains
cbs4_get_data("84287NED"
              , Perioden = contains(c("2019MM1", "2020"))
              , BedrijfstakkenBranchesSBI2008 = "T001081"
)

# filter on Perioden with contains or = "2019MM12
cbs4_get_data("84287NED"
              , Perioden = contains("2020") | "2019MM12"
              , BedrijfstakkenBranchesSBI2008 = "T001081"
)

# This all works on observations too
cbs4_get_observations( id        = "80784ned"     # table id
                     , Perioden  = "2019JJ00"     # Year 2019
                     , Geslacht  = "1100"         # code for total gender
                     , RegioS    = contains("PV") # provinces
                     , Measure   = "M003371_2"    # topic selection
                     )

# supply your own odata 4 query
cbs4_get_data("84287NED", query = "$filter=Perioden eq '2019MM12'")

# an odata 4 query will overrule other filter statements
cbs4_get_data("84287NED"
             , Perioden = "2018MM12"
             , query = "$filter=Perioden eq '2019MM12'"
             )

# With query argument an odata4 expression with other (filter) functions can be used
cbs4_get_observations(
  id     = "80784ned"    # table id
  ,query = paste0(       # odata4 query
     "$skip=4",          # skip the first 4 rows of the filtered result
     "&$top=20",         # then slice the first 20 rows of the filtered result
   "&$select=Measure,Geslacht,Perioden,RegioS,Value", # omit the Id and ValueAttribute fields
     "&$filter=endswith(Measure,'_1')") # filter only Measure ending on '_1'
  )

}
```

---

cbs4_search *Search table with search term*

Description

Search a opendata table using free text search.

Usage

```
cbs4_search(
  query,
  catalog = "CBS",
  language = "nl-nl",
  convert_dates = TRUE,
  verbose = getOption("cbsodata4.verbose", FALSE),
  base_url = getOption("cbsodata4.base_url", BASEURL4)
)
```

Arguments

query	character with the text to searched for
catalog	only show the datasets from that catalog. If NULL all datasets of all catalogs will be returned.
language	character language of the catalog, currently only Dutch
convert_dates	Converts date columns in Date-Time type (in stead of character)
verbose	Should the url request be printed?
base_url	base url of the CBS OData 4 API

Value

data.frame same format as cbs4_get_datasets() plus extra $rel column with the search score.

Note

The search engine currently searches in the odata3 data collection, but uses the Identifiers to find tables in the odata4 data collection.

Examples

```
if (interactive()){

  ds_nl <- cbs4_search("geboorte", language="nl-nl")
  ds_nl[1:3, c("Identifier", "Title", "rel")]

  bike_tables <- cbs4_search("fiets")
  bike_tables[1:10, c("Identifier", "Title", "rel")]
}
```

contains *Detect substring in column*

Description

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

Usage

```
contains(x, column = NULL, allowed = NULL)

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

Arguments

x	substring to be detected in column
column	column name
allowed	character with allowed values. If supplied it will check if x is a code in allowed.

See Also

Other odata4 query: eq()

Examples

```
if (interactive()){

  # filter on Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = "2019MM12" # december 2019
                )

  # filter on multiple Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = c("2019MM12", "2020MM01") # december 2019, january 2020
                )

  # to filter on a dimension just add the filter to the query

  # filter on Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = "2019MM12" # december 2019
                , BedrijfstakkenBranchesSBI2008 = "T001081"
                )
```

```
# filter on Perioden with contains
cbs4_get_data("84287NED"
              , Perioden = contains("2020")
              , BedrijfstakkenBranchesSBI2008 = "T001081"
)

# filter on Perioden with multiple contains
cbs4_get_data("84287NED"
              , Perioden = contains(c("2019MM1", "2020"))
              , BedrijfstakkenBranchesSBI2008 = "T001081"
)

# filter on Perioden with contains or = "2019MM12
cbs4_get_data("84287NED"
              , Perioden = contains("2020") | "2019MM12"
              , BedrijfstakkenBranchesSBI2008 = "T001081"
)

# This all works on observations too
cbs4_get_observations( id        = "80784ned"     # table id
                     , Perioden  = "2019JJ00"     # Year 2019
                     , Geslacht  = "1100"         # code for total gender
                     , RegioS    = contains("PV") # provinces
                     , Measure   = "M003371_2"    # topic selection
                     )

# supply your own odata 4 query
cbs4_get_data("84287NED", query = "$filter=Perioden eq '2019MM12'")

# an odata 4 query will overrule other filter statements
cbs4_get_data("84287NED"
              , Perioden = "2018MM12"
              , query = "$filter=Perioden eq '2019MM12'"
              )

# With query argument an odata4 expression with other (filter) functions can be used
cbs4_get_observations(
  id     = "80784ned"    # table id
  ,query = paste0(       # odata4 query
     "$skip=4",          # skip the first 4 rows of the filtered result
     "&$top=20",         # then slice the first 20 rows of the filtered result
   "&$select=Measure,Geslacht,Perioden,RegioS,Value", # omit the Id and ValueAttribute fields
     "&$filter=endswith(Measure,'_1')") # filter only Measure ending on '_1'
  )

}
```

---

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

```
eq(x, column = NULL, allowed = NULL)
```

Arguments

`x`	exact code(s) to be matched in `column`
`column`	name of column.
`allowed`	`character` with allowed values. If supplied it will check if `x` is a code in `allowed`.

Value

query object

See Also

Other odata4 query: `contains()`

Examples

```
if (interactive()){

  # filter on Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = "2019MM12" # december 2019
                )

  # filter on multiple Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = c("2019MM12", "2020MM01") # december 2019, january 2020
                )

  # to filter on a dimension just add the filter to the query

  # filter on Perioden (see meta$PeriodenCodes)
  cbs4_get_data("84287NED"
                , Perioden = "2019MM12" # december 2019
                , BedrijfstakkenBranchesSBI2008 = "T001081"
                )


  # filter on Perioden with contains
  cbs4_get_data("84287NED"
                 , Perioden = contains("2020")
                 , BedrijfstakkenBranchesSBI2008 = "T001081"
  )
```

```
  # filter on Perioden with multiple contains
  cbs4_get_data("84287NED"
                , Perioden = contains(c("2019MM1", "2020"))
                , BedrijfstakkenBranchesSBI2008 = "T001081"
  )

  # filter on Perioden with contains or = "2019MM12
  cbs4_get_data("84287NED"
                , Perioden = contains("2020") | "2019MM12"
                , BedrijfstakkenBranchesSBI2008 = "T001081"
  )

  # This all works on observations too
  cbs4_get_observations( id        = "80784ned"     # table id
                       , Perioden  = "2019JJ00"     # Year 2019
                       , Geslacht  = "1100"         # code for total gender
                       , RegioS    = contains("PV") # provinces
                       , Measure   = "M003371_2"    # topic selection
                       )

  # supply your own odata 4 query
  cbs4_get_data("84287NED", query = "$filter=Perioden eq '2019MM12'")

  # an odata 4 query will overrule other filter statements
  cbs4_get_data("84287NED"
                , Perioden = "2018MM12"
                , query = "$filter=Perioden eq '2019MM12'"
                )

  # With query argument an odata4 expression with other (filter) functions can be used
  cbs4_get_observations(
    id     = "80784ned"    # table id
    ,query = paste0(       # odata4 query
       "$skip=4",          # skip the first 4 rows of the filtered result
       "&$top=20",         # then slice the first 20 rows of the filtered result
     "&$select=Measure,Geslacht,Perioden,RegioS,Value", # omit the Id and ValueAttribute fields
       "&$filter=endswith(Measure,'_1')") # filter only Measure ending on '_1'
    )

}
```

Index