Package 'wiesbaden'

December 18, 2022

Title Access Databases from the Federal Statistical Office of Germany

Version 1.2.8

Date 2022-12-17

Author Moritz Marbach <m.marbach@ucl.ac.uk> [aut, cre]

Maintainer Moritz Marbach <m.marbach@ucl.ac.uk>

Description Retrieve and import data from different databases of the Federal Statistical Office of Germany (DESTATIS) using their SOAP XML web service <https://www-genesis.destatis.de/>.

Depends R (>= 3.3.1)

License GPL-3

URL https://github.com/sumtxt/wiesbaden/

BugReports https://github.com/sumtxt/wiesbaden/issues

Encoding UTF-8

Imports httr (>= 1.2.1), xml2 (>= 1.0.0), stringr (>= 1.1.0), stringi (>= 1.4.0), readr (>= 1.0.0), jsonlite (>= 1.6.0), keyring (>= 1.1.0)

RoxygenNote 7.2.1

Suggests knitr, rmarkdown, tidyverse

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2022-12-18 16:50:02 UTC

R topics documented:

`download_csv` *Download the csv-file of a table*

Description

`download_csv()` downloads the csv for a table

Usage

```
download_csv(
  tablename,
  startyear = "",
  endyear = "",
  ...,
  genesis_db = "de",
  save = TRUE
)
```

Arguments

`tablename`	name of the table to retrieve.
`startyear`	only retrieve values for years equal or larger to `startyear`. Default: "".
`endyear`	only retrieve values for years smaller or equal to `endyear`. Default: "".
`...`	further parameters supplied as URL parameter in the GENESIS database call
`genesis_db`	name of the database (default: 'de').
`save`	write string to a text file (default: TRUE)

Details

Downloads the csv file either to the working directory `getwd()` or outputs it as a string. This is an alternative approach to the retrieve_*() functions. This is designed for `https://www-genesis.destatis.de/genesis/online` as it does not require a login. It might not work as expected for the other databases.

See Also

`read_header_genesis`.

Examples

```
## Not run:

download_csv("12411-0004.csv")


## End(Not run)
```

`read_gv100` *Reads the DESTATIS GV100 Format*

Description

The GV100 format is used by DESTATIS to publish the German municipality register

Usage

```
read_gv100(file, stzrt, version = NULL, encoding = "iso-8859-1", ...)
```

Arguments

`file`	path to file
`stzrt`	integer to select the administrative level (see details)
`version`	which GV100 version. If NULL the version is guessed based on the file name.
`encoding`	encoding of the file
`...`	other parameters passed to `read_fwf`

Details

The Gemeindeverzeichnis (municipality register) is published in a fixed width file refered to as "GV1000 ASCII Format" by DESTATIS. The register features the list of municipality and higher order administrative units. The function is a wrapper around [readr::read_fwf()].

There are two types of files: One feature the administrative information (`version="AD"`) and one with non-administrative (`version="NAD"`). If `version=NULL`, read_gv100() guess the type based on the file name.

To select a particular administrative unit use the stzrt argument (Satzart). For the AD version, the following choices are possible:

10 - Länder (states) 20 - Regierungsbezirke 30 - Regionsdaten (only Baden-Württemberg) 40 - Kreise (counties) 50 - Gemeindeverbandsdaten 60 - Gemeinden (municipalities)

For the NAD version only:

41 - Kreise (counties) 61 - Gemeinden (municipalities)

Since about 2019, the Gemeindeverzeichnis is using UTF-8 encoding rather than ISO-8859-1.

Value

a data.frame.

See Also

https://www.destatis.de/DE/Themen/Laender-Regionen/Regionales/Gemeindeverzeichnis/_inhalt.html [readr::read_fwf()]

Examples

```
 ## Not run:

d <- read_gv100("GV100NAD31122016.asc", stzrt=60)


## End(Not run)
```

---

read_header_genesis *Read Header of a GENESIS csv*

Description

read_header_genesis reads the header of a GENESIS csv.

Usage

```
read_header_genesis(
  ...,
  start,
  lines = 2,
  readr_locale = locale(encoding = "windows-1252"),
  replacer = NULL,
  clean_letters = TRUE
)
```

Arguments

...	arguments to read_csv2
start	number of the first line of the header
lines	number of header lines
readr_locale	definition of locale() to be passed to read_csv2()
replacer	a vector that is used as the first K column-names
clean_letters	make proper variable names? (default: TRUE)
locale	default encoding is 'windows-1252'

Details

To generate valid column names, the function replaces all special characters (e.g. German öüä) with ASCII letters and removes whitespaces. Multi-line headers are joined but separated with a '_'.

Value

a `vector` of column names.

See Also

`read_csv2`

Examples

```
 ## Not run:

 library(readr)

 download_csv(tablename="12411-0004")

 d <- read_header_genesis('12411-0004.csv', start=6, replacer=c("STAG"))
 data <- read_csv2('12411-0004.csv', skip=6, n_max=30-6+1,
na="-", locale=locale(encoding="windows-1252") )
 colnames(data) <- d

## End(Not run)
```

`retrieve_data` *Retrieves Data from GENESIS Databases*

Description

`retrieve_data` retrieves a single data table.

Usage

```
retrieve_data(
  tablename,
  startyear = "",
  endyear = "",
  regionalmerkmal = "",
  regionalschluessel = "",
  sachmerkmal = "",
  sachschluessel = "",
```

```
  sachmerkmal2 = "",
  sachschluessel2 = "",
  sachmerkmal3 = "",
  sachschluessel3 = "",
  inhalte = "",
  genesis = NULL,
  language = "de",
  ...
)
```

Arguments

`tablename`	name of the table to retrieve.
`startyear`	only retrieve values for years equal or larger to `startyear`. Default: "".
`endyear`	only retrieve values for years smaller or equal to `endyear`. Default: "".
`regionalmerkmal`	key for Regionalklassifikation. See details for more information. Default: "".
`regionalschluessel`	only retrieve values for particular regional units. See details for more information. Default: "".
`sachmerkmal, sachmerkmal2, sachmerkmal3`	key for Sachklassifikation. Default: "".
`sachschluessel, sachschluessel2, sachschluessel3`	value for Sachklassifikation. Default: "".
`inhalte`	retrieve only selected variables. Default is to retrieve all.
`genesis`	to authenticate a user and set the database (see below).
`language`	retrieve information in German "de" (default) or in English "en" if available.
`...`	other arguments send to the httr::GET request.

Details

Use `retrieve_datalist` to find the `tablename` based on the table series you are interested in. See the package description (`wiesbaden`) for details about setting the login and database.

The parameter `regionalschluessel` can either be a single value (a single Amtlicher Gemeindeschlüssel) or a comma-separated list of values supplied as string (no whitespaces). Wildcard character "*" is allowed. If `regionalschluessel` is set, the parameter `regionalmerkmal` must also be set to GEMEIN, KREISE, REGBEZ, or DLAND. The same logic applies to the parameter combination `sachmerkmal` and `sachschluessel*`. The parameter `inhalte` takes a 1-6 character long name of a variable in the table. If choosing multiple variables, delimit by ",", e.g. "STNW01,STNW02" (no whitespaces).

Limiting the data request to particular years (via the `*year` parameters), geographical units (via the `regional*` parameters) attributes (via the `sach*` parameters) or selected variables (via the `inhalte` parameter) is necessary if the API request fails to return any data. If you are not able to download the table because of size, inspect the metadata first (using retrieve_metadata or retrieve_valuelabel) and then limit the data request accordingly. See also examples below.

Value

a `data.frame`. Value variables (_val) come with three additional variables (_qual, _lock, _err). The exact nature of these variables is unknown, but _qual appears to indicate if _val is a valid value. If _qual=="e" the value in _val is valid while if _qual!="e" (then _qual = ("-","/", ".", "x", ...)) it is typically zero should/might be set to NA.

See Also

`retrieve_datalist` `wiesbaden`

Examples

```
## Not run:
# Retrieve values for the table 14111KJ002 which contains the
# federal election results on the county level.
# Assumes that user/password are stored via save_credentials()

data <- retrieve_data(tablename="14111KJ002", genesis=c(db="regio") )

# ... only the values for the AfD.

data <- retrieve_data(tablename="14111KJ002", sachmerkmal="PART04",
  sachschluessel="AFD", genesis=c(db="regio") )

# ... or only values from Saxony

data <- retrieve_data(tablename="14111KJ002", regionalmerkmal="KREISE",
  regionalschluessel="14*", genesis=c(db="regio") )

# Limiting the number of data points is in particular important for
# large tables. For example, this data request fails:

data <- retrieve_data(tablename="33111GJ005", genesis=c(db='regio'))

# But after limiting the request to one year, the data is returned:

data <- retrieve_data(tablename="33111GJ005", genesis=c(db='regio'), startyear=2019, endyear=2019)

# An alternative strategy is to only request a subset of the variables.
# For example, this data request fails:

data <- retrieve_data("12711GJ002", genesis=c(db="regio"))

# But when requesting only one instead of all variables, the data is returned:

data <- retrieve_data("12711GJ002", inhalte="BEV081", genesis=c(db="regio"))



## End(Not run)
```

`retrieve_datalist` *Retrieves List of Tables from GENESIS Databases*

Description

`retrieve_datalist` retrieves a list of available data tables in a series.

Usage

```
retrieve_datalist(tableseries, genesis = NULL, language = "de", ...)
```

Arguments

`tableseries`	name of series for which tables should be retrieved.
`genesis`	to authenticate a user and set the database (see below).
`language`	retrieve information in German "de" (default) or in English "en" if available.
`...`	other arguments send to the httr::GET request.

Details

See the package description (`wiesbaden`) for details about setting the login and database. To retrieve a list of all available data use tableseries="*" or combine the wildcard character * with a prefix (see below for an example).

Value

a `data.frame`

See Also

`retrieve_data` `wiesbaden`

Examples

```
## Not run:
# Retrieves list of available tables for the table series 14111
# which contains the federal election results.
# Assumes that user/password are stored via save_credentials()

d <- retrieve_datalist(tableseries="14111*", genesis=c(db="regio") )

## End(Not run)
```

`retrieve_metadata` *Retrieves Meta Data from GENESIS Databases*

Description

`retrieve_metadata` retrieves meta data.

Usage

```
retrieve_metadata(tablename, language = "de", genesis = NULL, ...)
```

Arguments

`tablename`	name of the table to retrieve.
`language`	retrieve information in German "de" (default) or in English "en" if available.
`genesis`	to authenticate a user and set the database (see below).
`...`	other arguments send to the httr::GET request.

Details

See the package description (`wiesbaden`) for details about setting the login and database.

Value

a `data.frame`.

See Also

`wiesbaden`

Examples

```
## Not run:
# Meta data contain the explanations to the variable names for the table
# federal election results on the county level.
# Assumes that user/password are stored via save_credentials()

metadata <- retrieve_metadata(tablename="14111KJ002", genesis=c(db="regio") )

## End(Not run)
```

retrieve_valuelabel *Retrieves Value Labels from GENESIS Databases*

Description

retrieve_valuelabel retrieves value labels for variable

Usage

```
retrieve_valuelabel(
  variablename,
  valuelabel = "*",
  genesis = NULL,
  language = "de",
  ...
)
```

Arguments

variablename	name of the variable
valuelabel	"*" (default) retrieves all value labels.
genesis	to authenticate a user and set the database (see below).
language	retrieve information in German "de" (default) or in English "en" if available.
...	other arguments send to the httr::GET request.

Details

See the package description (wiesbaden) for details about setting the login and database.

Value

a data.frame.

See Also

retrieve_datalist wiesbaden

Examples

```
## Not run:
# Value labels contain for the variable 'PART04' in the table with the
# federal election results on the county level.
# Assumes that user/password are stored via save_credentials()

metadata <- retrieve_valuelabel(variablename="PART04", genesis=c(db="regio") )
```

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

`retrieve_varinfo` *Retrieves further information on a variable from GENESIS Databases*

Description

`retrieve_varinfo` retrieves further information.

Usage

```
retrieve_varinfo(variablename, genesis = NULL, language = "de", ...)
```

Arguments

`variablename`	name of the variable
`genesis`	to authenticate a user and set the database (see below).
`language`	retrieve information in German "de" (default) or in English "en" if available.
`...`	other arguments send to the httr::GET request.

Details

See the package description (`wiesbaden`) for details about setting the login and database.

Value

a `data.frame`.

See Also

`retrieve_datalist` `wiesbaden`

Examples

```
 ## Not run:
 # Variable information 'AI2105' (Anteil der Empfänger von Arbeitslosengeld II im Alter
 # von 15 bis 24 Jahren an der Bevölkerung gleichen Alters)
 # Assumes that user/password are stored via save_credentials()

 metadata <- retrieve_varinfo(variablename="AI2105", genesis=c(db="regio") )

## End(Not run)
```

save_credentials *Saves database credentials*

Description

save_credentials saves a set of database credentials using the keyring package.

Usage

```
save_credentials(db, user, password)
```

Arguments

db	database name, either 'nrw', 'regio', 'de' or 'bm'.
user	your user name.
password	your password.

Details

User/password are stored in Keychain on macOS, Credential Store on Windows or Secret Service API on Linux. If a user/password pair for a database already exists, it is silently replaced with the new pair. This function relies on the keyring package.

See Also

wiesbaden, keyring

test_login *Tests Login in GENESIS Databases*

Description

test_login tests if the login works.

Usage

```
test_login(genesis = NULL, ...)
```

Arguments

genesis	to authenticate a user and set the database (see below).
...	other arguments send to the httr::GET request.

Value

a `string` with the server return message.

Examples

```
## Not run:

test_login(genesis=c(db="regio") )


## End(Not run)
```

Index