Package 'EDCimport'

May 19, 2023

Version 0.3.0

Title Import Data from EDC Software

Description A convenient toolbox to import data exported from Electronic Data Capture (EDC) software 'TrialMaster' and 'Macro'.

License GPL-3

URL https://github.com/DanChaltiel/EDCimport

BugReports https://github.com/DanChaltiel/EDCimport/issues

Depends R (>= 3.1.0)

Imports cli, dplyr, forcats, glue, ggplot2, haven, labelled, purrr, readr, rlang, stringr, tibble, tidyr, tidyselect

Suggests callr, crosstable, gtools, janitor, knitr, plotly, testthat (>= 3.1.8), vdiffr, withr

Encoding UTF-8

RoxygenNote 7.2.3

Config/testthat/edition 3

NeedsCompilation no

Author Dan Chaltiel [aut, cre] (<https://orcid.org/0000-0003-3488-779X>)

Maintainer Dan Chaltiel <dan.chaltiel@gmail.com>

Repository CRAN

Date/Publication 2023-05-19 10:30:02 UTC

R topics documented:

`data_example` *Example databases*

Description

List of tables used in EDCimport examples.

Usage

```
edc_example_mixed(N = 100)

edc_example_plot(N = 50, seed = 42)
```

Arguments

`N`	the number of patients
`seed`	the random seed

Value

a list of tables

`EDCimport_options` *Options used in* `EDCimport`

Description

Here are all the options used in `EDCimport`:

- `trialmaster_pw`: the password for the archive read by `read_trialmaster()`
- `edc_lookup`: the default data used by `find_keyword()`
- `path_7zip`: the path to 7-zip directory if not in the PATH, usually "C:/Program Files/7-Zip/" on Windows. See `extract_7z()` for more information.

edc_swimmerplot *Swimmer plot of all dates columns*

Description

Join all tables from `.lookup$dataset` on `id`

Usage

```
edc_swimmerplot(
  .lookup = getOption("edc_lookup", NULL),
  ...,
  id = "SUBJID",
  group = NULL,
  origin = NULL,
  time_unit = c("days", "weeks", "months", "years"),
  aes_color = c("variable", "label"),
  plotly = TRUE
)
```

Arguments

`.lookup`	the lookup table, loaded along with the database or result of `get_lookup()`
`...`	not used
`id`	the patient identifier
`group`	a grouping variable, given as "dataset$column"
`origin`	a variable to consider as time 0, given as "dataset$column"
`time_unit`	if `origin!=NULL`, the unit to measure time. One of `c("days", "weeks", "months", "years")`.
`aes_color`	either `variable` ("dataset - column") or `label` (the column label)
`plotly`	whether to use {`plotly`} to get an interactive plot

Value

either a `plotly` or a `ggplot`

Examples

```
#tm = read_trialmaster("filename.zip", pw="xx")
tm = edc_example_plot()
load_list(tm)
p = edc_swimmerplot(.lookup)
p2 = edc_swimmerplot(.lookup, origin="db0$date_naissance", time_unit="weeks")
p3 = edc_swimmerplot(.lookup, group="db0$group", aes_color="label")
## Not run:
#save the plotly plot as HTML to share it
```

```
htmlwidgets::savewidget(p, "edc_swimmerplot.html", selfcontained=TRUE)

## End(Not run)
```

`extend_lookup` *Extend the lookup table*

Description

This utility extends the lookup table to include:

- `n_id` the number of patients present in the dataset
- `rows_per_id` the mean number of row per patient
- `crfname` the actual name of the dataset

Usage

```
extend_lookup(
  lookup,
  ...,
  key_columns = get_key_cols(),
  datasets = get_datasets(lookup)
)
```

Arguments

`lookup`	[`data.frame(1)`] the lookup table
`...`	unused
`key_columns`	[`list`] the result of `get_key_cols()`, containing the column name used for patient ID and CRF name. Important for `split_mixed` and `extend_lookup`.
`datasets`	[`data.frame(n)`] for experts only

Value

the lookup, extended

Examples

```
#tm = read_trialmaster("filename.zip", pw="xx")
tm = edc_example_mixed()
load_list(tm)
.lookup
.lookup = extend_lookup(.lookup)
.lookup
```

`extract_7z` *Extract an archive using 7zip*

Description

Use 7zip to extract an archive. Obviously, 7zip should be installed beforehand.

Usage

```
extract_7z(archive, target_dir, password = NULL, path_7zip = NULL)
```

Arguments

`archive`	the archive file
`target_dir`	the target directory
`password`	the password of the archive, if any
`path_7zip`	See section below. Default to `get_7z_dir()`.

Value

the success/error message. Mainly used for its side effect of extracting the archive.

Install 7-zip

If 7-zip is not installed on your computer, the easiest way to get it is to run `installr::install.7zip()`.

Add 7zip to the `PATH`

For this function to work, 7-zip should be registered as a `PATH` environment variable. There are many resources online to help you register if, e.g. https://www.java.com/en/download/help/path.html. If you cannot change the `PATH` on your computer, then you can add the path to 7zip executable directory in the parameter `path_7zip` or globally using `options(path_7zip="path/to/7zip")`. This will add 7zip to the `PATH` for the current session.

See Also

https://info.nrao.edu/computing/guide/file-access-and-archiving/7zip/7z-7za-command-line-guide#section-17

`find_keyword` *Find a keyword*

Description

Find a keyword in all names and labels of a list of datasets.

Usage

```
find_keyword(keyword, data = getOption("edc_lookup", NULL), ignore_case = TRUE)
```

Arguments

`keyword`	the keyword to search for
`data`	the dataframe where to search the keyword. Can be set using `options(edc_lookup=my_data)`, which is done automatically when calling `read_trialmaster()`.
`ignore_case`	should case differences be ignored in the match?

Value

a filtered tibble

EDCimport

Usually,

Examples

```
library(crosstable)
library(dplyr)
library(purrr)

#Using a custom table list
lookup = list(i=crosstable::iris2, m=crosstable::mtcars2) %>% get_lookup()
find_keyword("hp", data=lookup)
find_keyword("number|date", data=lookup)
find_keyword("number|date", data=lookup, ignore_case=FALSE)

#Using lookup from [read_trialmaster()]
## Not run:
path = system.file("extdata/Example_Export_SAS_XPORT_2022_08_25_15_16.zip",
                   package = "EDCimport", mustWork=TRUE)
w = read_trialmaster(path, verbose=FALSE)
options(edc_lookup=w$.lookup) #optional
find_keyword("patient")

## End(Not run)
```

get_7z_dir *Retrieve the path to 7zip executable directory*

Description

Retrieve the path to 7zip executable directory. Use `options(path_7zip="path/to/7zip")` to change its behavior.

Usage

```
get_7z_dir()
```

Value

the path to 7zip executable directory

get_datasets *Retrieve the datasets*

Description

Get the datasets from the lookup table as a list of data.frames.

Usage

```
get_datasets(lookup = getOption("edc_lookup", NULL), envir = parent.frame())
```

Arguments

`lookup`	the lookup table
`envir`	(internal use)

Value

a list of all datasets

get_key_cols *Important column names*

Description

Important column names

Usage

```
get_key_cols(
  patient_id = getOption("edc_id", c("ptno", "subjid")),
  crfname = getOption("edc_crfname", "crfname"),
  ...
)
```

Arguments

patient_id	the name of the columns containing the patient ID
crfname	the name of the columns containing the name of the CRF page
...	unused

get_lookup *Generate a lookup table*

Description

Generate a lookup table

Usage

```
get_lookup(data_list)
```

Arguments

data_list	a list containing at least 1 dataframe

Value

a dataframe summarizing column names and labels

Examples

```
tl = list(r=crosstable::iris2, x=mtcars) %>% get_lookup()
tl
library(tidyr)
tl %>% unnest(everything()) %>% unnest(everything())
```

load_as_list *Load a* .RData *file as a list*

Description

Instead of loading a .RData file in the global environment, extract every object into a list.

Usage

```
load_as_list(filename)
```

Arguments

filename	the filename, with the .RData extension.

Value

a list

Examples

```
x = list(a=1, b=mtcars)
save_list(x, "test.RData")
y = load_as_list("test.RData")
print(y$a)
```

load_list *Load a list in an environment*

Description

Load a list in an environment

Usage

```
load_list(x, env = parent.frame(), remove = TRUE)
```

Arguments

x	a list
env	the environment onto which the list should be loaded
remove	if TRUE, x will be removed from the environment afterward

Value

nothing, called for its side-effect

Examples

```
x=list(a=1, b=mtcars)
load_list(x, remove=FALSE)
print(a)
print(nrow(b))
```

`read_tm_all_xpt` *Read all* `.xpt` *files in a directory*

Description

Read all `.xpt` files in a directory (unzipped TrialMaster archive).
If `7zip` is installed, you should probably rather use `read_trialmaster()` instead.
If a `procformat.sas` file exists in the directory, formats will be applied.

Usage

```
read_tm_all_xpt(
  directory,
  ...,
  format_file = "procformat.sas",
  clean_names_fun = NULL,
  split_mixed = FALSE,
  extend_lookup = TRUE,
  key_columns = get_key_cols(),
  datetime_extraction = NULL
)
```

Arguments

`directory`	[character(1)] the path to the unzipped archive using SAS_XPORT format. Will read the extraction date from the directory name.
`...`	unused
`format_file`	[character(1)] the path to the `procformat.sas` file that should be used to apply formats. Use `NULL` to not apply formats.
`clean_names_fun`	[function] a function to clean column names, e.g. `janitor::clean_names()`
`split_mixed`	[logical(1): FALSE] whether to split mixed datasets. See split_mixed_datasets.
`extend_lookup`	[character(1): FALSE] whether to enrich the lookup table. See extend_lookup.

key_columns [list]
the result of get_key_cols(), containing the column name used for patient ID and CRF name. Important for split_mixed and extend_lookup.

datetime_extraction
[POSIXt(1)]
the datetime of the data extraction. Default to the most common date of last modification in directory.

Value

a list containing one dataframe for each .xpt file in the folder, the extraction date (datetime_extraction), and a summary of all imported tables (.lookup). If not set yet, option edc_lookup is automatically set to .lookup.

---

read_trialmaster *Read the .zip archive of a TrialMaster export*

Description

Import the .zip archive of a TrialMaster trial export as a list of dataframes. The archive filename should be leaved untouched as it contains the project name and the date of extraction.
Generate a .rds cache file for future reads.
If 7zip is not installed or available, use read_tm_all_xpt() instead.

Usage

```
read_trialmaster(
  archive,
  ...,
  use_cache = TRUE,
  clean_names_fun = NULL,
  split_mixed = FALSE,
  extend_lookup = FALSE,
  key_columns = get_key_cols(),
  pw = getOption("trialmaster_pw"),
  verbose = getOption("edc_verbose", 1)
)
```

Arguments

archive [character(1)]
the path to the archive

... unused

use_cache [logical(1): 'TRUE']
if TRUE, read the .rds cache if any or extract the archive and create a cache. If FALSE extract the archive without creating a cache file.

`clean_names_fun`
: [function]
a function to clean column names, e.g. `janitor::clean_names()`

`split_mixed`
: [logical(1): FALSE]
whether to split mixed datasets. See split_mixed_datasets.

`extend_lookup`
: [character(1): FALSE]
whether to enrich the lookup table. See extend_lookup.

`key_columns`
: [list]
the result of `get_key_cols()`, containing the column name used for patient ID and CRF name. Important for `split_mixed` and `extend_lookup`.

`pw`
: [character(1)]
The password if the archive is protected. To avoid writing passwords in plain text, it is probably better to use `options(trialmaster_pw="xxx")` instead though.

`verbose`
: [logical(1)]
one of `c(0, 1, 2)`. The higher, the more information will be printed.

Value

a list containing one dataframe for each `.xpt` file in the folder, the extraction date (`datetime_extraction`), and a summary of all imported tables (`.lookup`). If not set yet, option `edc_lookup` is automatically set to `.lookup`.

`save_list` *Save a list as* `.RData` *file*

Description

Save a list as `.RData` file

Usage

```
save_list(x, filename)
```

Arguments

`x`
: a list

`filename`
: the filename, with the `.RData` extension.

Value

nothing, called for its side-effect

Examples

```
x=list(a=1, b=mtcars)
save_list(x, "test.RData")
load("test.RData")
file.remove("test.RData")
print(a)
print(nrow(b))
```

`split_mixed_datasets` *Split mixed datasets*

Description

Split mixed tables, i.e. tables that hold both long data (N values per patient) and short data (one value per patient, duplicated on N lines), into one long table and one short table.

Usage

```
split_mixed_datasets(
  datasets = get_datasets(),
  id,
  ...,
  ignore_cols = getOption("edc_crfname", "CRFNAME"),
  output_code = FALSE,
  verbose = TRUE
)
```

Arguments

`datasets`	the datasets to consider. Use the helper `get_datasets()` if needed.
`id`	the patient identifier, probably "SUBJID". Should be shared by all datasets.
`...`	not used
`ignore_cols`	columns to ignore when considering a table as long. default to
`output_code`	whether to print the code to explicitly write. Can also be a file path.
`verbose`	whether to print informations about the process.

Value

a list of the new long and short tables. Use `load_list()` to load them into the global environment.

Examples

```
#tm = read_trialmaster("filename.zip", pw="xx")
tm = edc_example_mixed()
names(tm)
#load_list(tm)
print(tm$long_mixed) #`val1` and `val2` are long but `val3` is short

mixed_data = split_mixed_datasets(tm, id="SUBJID", verbose=TRUE)
load_list(mixed_data)
print(long_mixed_short)
print(long_mixed_long)

#alternatively, get the code and only use the datasets you need
split_mixed_datasets(tm, id="SUBJID", output_code=TRUE)
filename = tempfile("mixed_code", fileext=".R")
split_mixed_datasets(tm, id="SUBJID", output_code=filename)
readLines(filename)
```

---

unify *Unify a vector*

Description

Turn a vector of length N to a vector of length 1 after checking that there is only one unique value. Useful to safely flatten a duplicated table. This preserves the `label` attribute if set.

Usage

```
unify(x)
```

Arguments

x	a vector

Value

a vector of length 1

Examples

```
unify(c(1,1,1,1))
#unify(c(1,1,2,1)) #warning

library(dplyr)
x=tibble(id=rep(letters[1:5],10), value=rep(1:5,10))
x %>% group_by(id) %>% summarise(value=unify(value)) #safer than `value=value[1]`
x$value[2]=1
#x %>% group_by(id) %>% summarise(value=unify(value)) #warning about that non-unique value
```

Index