Package ‘xportr’

October 14, 2022

Title Utilities to Output CDISC SDTM/ADaM XPT Files

Version 0.1.0

Description Tools to build CDISC compliant data sets and check for CDISC compliance.

URL https://github.com/atorus-research/xportr

BugReports https://github.com/atorus-research/xportr/issues

Imports dplyr (>= 1.0.2), purrr (>= 0.3.4), stringr (>= 1.4.0),
          magrittr, glue (>= 1.4.2), rlang (>= 0.4.10), cli, tidyselect,
          readr, janitor, tm, haven (>= 2.5.0)

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.0

Suggests testthat (>= 3.0.0), withr, knitr, rmarkdown, readxl, DT,
          labelled, admiral, devtools, spelling, usethis, lintr, styler

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Eli Miller [aut, cre] (<https://orcid.org/0000-0002-2127-9456>),
          Vignesh Thanikachalam [aut],
          Ben Straub [aut],
          Ross Didenko [aut],
          Atorus/GSK JPT [cph]

Maintainer Eli Miller <Eli.Miller@AtorusResearch.com>

Repository CRAN

Date/Publication 2022-06-21 09:00:02 UTC

R topics documented:

   label_log .......................................................... 2
   length_log .......................................................... 2
   type_log ............................................................ 3
**Description**
Utility for Variable Labels

**Usage**

```
label_log(miss_vars, verbose)
```

**Arguments**

- `miss_vars`  
  Missing variables in metadata
- `verbose`  
  Provides additional messaging for user

**Value**

Output to Console

**Description**
Utility for Lengths

**Usage**

```
length_log(miss_vars, verbose)
```
**type_log**

**Arguments**
- **miss_vars** Variables missing from metadata
- **verbose** Provides additional messaging for user

**Value**
Output to Console

---

**var_names_log**

**Description**
Utility for Renaming Variables

**Usage**
var_names_log(tidy_names_df, verbose)

**Arguments**
- **tidy_names_df** dataframe
- **verbose** Provides additional messaging for user
**xportr_df_label**

### Description

Assigns dataset label from a dataset level metadata to a given data frame.

### Usage

```r
xportr_df_label(.df, metacore, domain = NULL)
```

### Arguments

- **.df**
  
  A data frame of CDISC standard.

- **metacore**
  
  A data frame containing dataset level metadata.

- **domain**
  
  A character value to subset the `.df`. If NULL (default), uses `.df` value as a subset condition.

### Value

Data frame with label attributes.
xportr_format

See Also

xportr_label(), xportr_format() and xportr_length()

Other metadata functions: xportr_format(), xportr_label(), xportr_length()

Examples

```r
adsl <- data.frame(
  USUBJID = c(1001, 1002, 1003),
  SITEID = c(001, 002, 003),
  AGE = c(63, 35, 27),
  SEX = c("M", "F", "M")
)

metacore <- data.frame(
  dataset = c("adsl", "adae"),
  label = c("Subject-Level Analysis", "Adverse Events Analysis")
)

adsl <- xportr_df_label(adsl, metacore)
```

---

**Description**

Assigns a SAS format from a variable level metadata to a given data frame.

**Usage**

```r
xportr_format(
  .df, 
  metacore, 
  domain = NULL, 
  verbose =getOption("xportr.format_verbose", "none")
)
```

**Arguments**

- `.df` A data frame of CDISC standard.
- `metacore` A data frame containing variable level metadata.
- `domain` A character value to subset the `.df`. If NULL(default), uses `.df` value as a subset condition.
- `verbose` The action the function takes when a variable label isn’t found. Options are ‘stop’, ‘warn’, ‘message’, and ‘none’

**Value**

Data frame with SAS format attributes for each variable.
See Also

- `xportr_label()`, `xportr_df_label()` and `xportr_length()`

Other metadata functions: `xportr_df_label()`, `xportr_label()`, `xportr_length()`

Examples

```r
adsl <- data.frame(
  USUBJID = c(1001, 1002, 1003),
  BRTHDT = c(1, 1, 2)
)

metacore <- data.frame(
  dataset = c("adsl", "adsl"),
  variable = c("USUBJID", "BRTHDT"),
  format = c(NA, "DATE9.")
)

adsl <- xportr_format(adsl, metacore)
```

---

**xportr_label**

*Assign Variable Label*

Description

Assigns variable label from a variable level metadata to a given data frame.

Usage

```r
xportr_label(
  .df, 
  metacore, 
  domain = NULL, 
  verbose = getOption("xportr.label_verbose", "none")
)
```

Arguments

- `.df` A data frame of CDISC standard.
- `metacore` A data frame containing variable level metadata.
- `domain` A character value to subset the `.df`. If NULL(default), uses `.df` value as a subset condition.
- `verbose` The action the function takes when a variable length isn’t Found. Options are 'stop', 'warn', 'message', and 'none'

Value

Data frame with label attributes for each variable.
**xportr_length**

See Also

`xportr_df_label()`, `xportr_format()` and `xportr_length()`

Other metadata functions: `xportr_df_label()`, `xportr_format()`, `xportr_length()`

Examples

```r
adsl <- data.frame(
  USUBJID = c(1001, 1002, 1003),
  SITEID = c(001, 002, 003),
  AGE = c(63, 35, 27),
  SEX = c("M", "F", "M")
)

metacore <- data.frame(
  dataset = "adsl",
  variable = c("USUBJID", "SITEID", "AGE", "SEX"),
  label = c("Unique Subject Identifier", "Study Site Identifier", "Age", "Sex")
)

adsl <- xportr_label(adsl, metacore)
```

---

**xportr_length**

Assign SAS Length

Description

Assigns SAS length from a variable level metadata to a given data frame.

Usage

```r
xportr_length(
  .df,
  metacore,
  domain = NULL,
  verbose = getOption("xportr.length_verbose", "none")
)
```

Arguments

- `.df` A data frame of CDISC standard.
- `metacore` A data frame containing variable level metadata.
- `domain` A character value to subset the `.df`. If `NULL` (default), uses `.df` value as a subset condition.
- `verbose` The action the function takes when a length isn’t found in metadata. Options are 'stop', 'warn', 'message', and 'none'
xportr_logger

Value

Data frame with SASlength attributes for each variable.

See Also

xportr_label(), xportr_df_label() and xportr_format()

Other metadata functions: xportr_df_label(), xportr_format(), xportr_label()

Examples

adsl <- data.frame(
    USUBJID = c(1001, 1002, 1003),
    BRTHDT = c(1, 1, 2)
)

metacore <- data.frame(
    dataset = c("adsl", "adsl"),
    variable = c("USUBJID", "BRTHDT"),
    length = c(10, 8)
)

adsl <- xportr_length(adsl, metacore)

xportr_logger

Utility Logging Function

Description

Functions to output user messages, usually relating to differences found between dataframe and the metacore/metadata object

Usage

xportr_logger(message, type = "none", ...)

Arguments

message Output to be sent out for user
type Three types: abort, warn, inform
... additional arguments if needed

Value

Output to Console
xportr_order

Order variables of a dataset according to Spec

Description

Order variables of a dataset according to Spec

Usage

xportr_order(
  .df,
  metacore,
  domain = NULL,
  verbose = getOption("xportr.order_verbose", "none")
)

Arguments

  .df         A data frame of CDISC standard.
  metacore    A data frame containing variable level metadata.
  domain      A character value to subset the .df. If NULL(default), uses .df value as a subset condition.
  verbose     Option for messaging order results

Value

Dataframe that has been re-ordered according to spec

xportr_type

Coerce variable type

Description

Current assumptions: columns_meta is a data.frame with names "Variables", "Type"

Usage

xportr_type(
  .df,
  metacore,
  domain = NULL,
  verbose = getOption("xportr.type_verbose", "none")
)
xportr_write

Arguments

- `.df` An R object with columns that can be coerced
- `metacore` Either a data.frame that has the names of all possible columns and their types, or a Metacore object from the Metacore package. Required column names are `dataset`, `variables`, `type`
- `domain` Name of the dataset. Ex ADAE/DM. This will be used to subset the metacore object. If none is passed it is assumed to be the name of the dataset passed in `.df`
- `verbose` The action the function takes when a variable isn’t typed properly. Options are 'stop', 'warn', 'message', and 'none'

Value

Returns the modified table.

Examples

```r
metacore <- data.frame(
  dataset = "test",
  variable = c("Subj", "Param", "Val", "NotUsed"),
  type = c("numeric", "character", "numeric", "character")
)

.df <- data.frame(
  Subj = as.character(123, 456, 789),
  Different = c("a", "b", "c"),
  Val = c("1", "2", "3"),
  Param = c("param1", "param2", "param3")
)

df2 <- xportr_type(.df, metacore, "test")
```

---

**xportr_write**  
*Write xpt v5 transport file*

Description

Writes a local data frame into SAS transport file of version 5. The SAS transport format is an open format, as is required for submission of the data to the FDA.

Usage

```r
xportr_write(.df, path, label = NULL)
```
Arguments

- **df**: A data frame to write.
- **path**: Path where transport file will be written. File name sans will be used as xpt name.
- **label**: Dataset label. It must be ≤40 characters.

Details

- Variable and dataset labels are stored in the "label" attribute.
- SAS length are stored in the "SASlength" attribute.
- SAS format are stored in the "SASformat" attribute.
- SAS type are stored in the "SASType" attribute.

Value

A data frame. `xportr_write()` returns the input data invisibly.
Index

* metadata functions
  xportr_df_label, 4
  xportr_format, 5
  xportr_label, 6
  xportr_length, 7

label_log, 2
length_log, 2

type_log, 3

var_names_log, 3
var_ord_msg, 4

xportr_df_label, 4, 6–8
xportr_df_label(), 6–8
xportr_format, 5, 5, 7, 8
xportr_format(), 5, 7, 8
xportr_label, 5, 6, 6, 8
xportr_label(), 5, 6, 8
xportr_length, 5–7, 7
xportr_length(), 5–7
xportr_logger, 8
xportr_order, 9
xportr_type, 9
xportr_write, 10