Package 'dmtools'

September 1, 2020

Title Tools for Validation the Dataset

Version 0.2.5

Description For checking the dataset from EDC(Electronic Data Capture) in clinical trials.
'dmtools' can check laboratory, dates and rename the dataset.
Laboratory - does the investigator correctly estimate the laboratory analyzes?
Dates - do all dates correspond to the protocol's timeline?
If the clinical trial has different lab reference ranges, 'dmtools' also can help.

Depends R (>= 3.6)

Imports magrittr (>= 1.5), dplyr (>= 1.0.0), readxl (>= 1.3.1), purrr (>= 0.3.3), lubridate (>= 1.7.4), httr (>= 1.4.1), tidyr (>= 1.1.0), tibble (>= 3.0.1)

License MIT + file LICENSE

URL https://github.com/chachabooms/dmtools

BugReports https://github.com/chachabooms/dmtools/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Konstantin Ryabov [aut, cre]

Maintainer Konstantin Ryabov <chachabooms@gmail.com>

Repository CRAN

Date/Publication 2020-09-01 12:40:03 UTC

R topics documented:

`add_cols` *Add columns if columns don't exist*

Description

Add columns if columns don't exist

Usage

```
add_cols(dset, ds_part, target_cols)
```

Arguments

`dset`	A data frame. The dataset.
`ds_part`	A character scalar. Prefix or postfix.
`target_cols`	A character vector with necessary columns.

Value

A data frame. The dataset.

`calc_diff` *Function for calculating the difference between two dates*

Description

Function for calculating the difference between two dates

Usage

```
calc_diff(st_inter, dt_item)
```

Arguments

`st_inter`	An interval. An object of interval.
`dt_item`	A date item. An object of date.

Value

An integer scalar. Differences between the two dates.

`check` *Check*

Description

Check

Usage

```
check(obj, dataset)
```

Arguments

`obj`	An object for check.
`dataset`	A dataset, a type is a data frame.

Value

An object with a check result.

Examples

```
id <- c("01", "02", "03")
screen_date_E1 <- c("1991-03-13", "1991-03-07", "1991-03-08")
rand_date_E2 <- c("1991-03-15", "1991-03-11", "1991-03-10")
ph_date_E3 <- c("1991-03-21", "1991-03-16", "1991-03-16")
bio_date_E3 <- c("1991-03-23", "1991-03-16", "1991-03-16")

df <- data.frame(id, screen_date_E1, rand_date_E2, ph_date_E3, bio_date_E3,
  stringsAsFactors = FALSE
)

timeline <- system.file("dates.xlsx", package = "dmtools")
obj_date <- date(timeline, id, dplyr::contains)

obj_date <- check(obj_date, df)
```

`check.default` *Check*

Description

Check

Usage

```
## Default S3 method:
check(obj, dataset)
```

Arguments

`obj`	An object for check.
`dataset`	A dataset, a type is a data frame.

Value

An object with a check result.

Examples

```
id <- c("01", "02", "03")
screen_date_E1 <- c("1991-03-13", "1991-03-07", "1991-03-08")
rand_date_E2 <- c("1991-03-15", "1991-03-11", "1991-03-10")
ph_date_E3 <- c("1991-03-21", "1991-03-16", "1991-03-16")
bio_date_E3 <- c("1991-03-23", "1991-03-16", "1991-03-16")

df <- data.frame(id, screen_date_E1, rand_date_E2, ph_date_E3, bio_date_E3,
  stringsAsFactors = FALSE
)
```

```
timeline <- system.file("dates.xlsx", package = "dmtools")
obj_date <- date(timeline, id, dplyr::contains)

obj_date <- check(obj_date, df)
```

`check_sites` *Check sites*

Description

Check sites

Usage

```
check_sites(objs, dataset, col_site)
```

Arguments

`objs`	A list of objects.
`dataset`	A dataset, a type is a data frame.
`col_site`	A column name of a site in the dataset, without quotes.

Value

A list of objects with a check result.

Examples

```
site <- c("site 01", "site 02")
id <- c("01", "02")
age <- c("19", "20")
sex <- c("f", "m")
gluc_post <- c("5.5", "4.1")
gluc_res_post <- c("norm", "no")
ast_post <- c("30", "48")
ast_res_post <- c(NA, "norm")

df <- data.frame(
  site, id, age, sex,
  gluc_post, gluc_res_post,
  ast_post, ast_res_post,
  stringsAsFactors = FALSE
)

refs_s01 <- system.file("labs_refer_s01.xlsx", package = "dmtools")
refs_s02 <- system.file("labs_refer_s02.xlsx", package = "dmtools")

s01_lab <- lab(refs_s01, id, age, sex, "norm", "no", site = "site 01")
s02_lab <- lab(refs_s02, id, age, sex, "norm", "no", site = "site 02")
```

```
labs <- list(s01_lab, s02_lab)
labs <- check_sites(labs, df, site)
```

`choose_test` *Filter final result*

Description

Filter final result

Usage

```
choose_test(obj, test, group_id)
```

Arguments

`obj`	An object for check.
`test`	Parameters, which use to filter the final dataset.
`group_id`	A logical scalar, default is TRUE.True is grouped by id, otherwise, it isn't grouped.

Value

The filtered dataset.

Examples

```
id <- c("01", "02", "03")
screen_date_E1 <- c("1991-03-13", "1991-03-07", "1991-03-08")
rand_date_E2 <- c("1991-03-15", "1991-03-11", "1991-03-10")
ph_date_E3 <- c("1991-03-21", "1991-03-16", "1991-03-16")
bio_date_E3 <- c("1991-03-23", "1991-03-16", "1991-03-16")

df <- data.frame(id, screen_date_E1, rand_date_E2, ph_date_E3, bio_date_E3,
  stringsAsFactors = FALSE
)

timeline <- system.file("dates.xlsx", package = "dmtools")
obj_date <- date(timeline, id, dplyr::contains)

obj_date <- check(obj_date, df)
choose_test(obj_date, "out")
```

choose_test.date *Filter final result*

Description

Filter final result

Usage

```
## S3 method for class 'date'
choose_test(obj, test = "out", group_id = T)
```

Arguments

obj	An object for calculation. Class date.
test	A character scalar. Parameters, which use to filter the final dataset, default: "out": "out" - dates, which are out of the protocol's timeline, "uneq" - dates, which are unequal, "ok" - correct dates, "skip" - empty dates.
group_id	A logical scalar, default is TRUE.True is grouped by id, otherwise, it isn't grouped.

Value

The dataset by a value of test.

Examples

```
id <- c("01", "02", "03")
screen_date_E1 <- c("1991-03-13", "1991-03-07", "1991-03-08")
rand_date_E2 <- c("1991-03-15", "1991-03-11", "1991-03-10")
ph_date_E3 <- c("1991-03-21", "1991-03-16", "1991-03-16")
bio_date_E3 <- c("1991-03-23", "1991-03-16", "1991-03-16")

df <- data.frame(id, screen_date_E1, rand_date_E2, ph_date_E3, bio_date_E3,
  stringsAsFactors = FALSE
)

timeline <- system.file("dates.xlsx", package = "dmtools")
obj_date <- date(timeline, id, dplyr::contains)

obj_date <- check(obj_date, df)
choose_test(obj_date, "out")
```

choose_test.lab *Filter final result*

Description

Filter final result

Usage

```
## S3 method for class 'lab'
choose_test(obj, test = "mis", group_id = T)
```

Arguments

obj	An object. Class lab.
test	A character scalar. Parameters, which use to filter the final dataset, default: "mis": "ok" - analysis, which has a correct estimate of the result, "mis" - analysis, which has an incorrect estimate of the result, "skip" - analysis, which has an empty value of the estimate, "null" - analysis, which has an empty result and value of the estimate.
group_id	A logical scalar, default is TRUE.True is grouped by id, otherwise, it isn't grouped.

Value

The filtered dataset by a value of test.

Examples

```
id <- c("01", "02", "03")
site <- c("site 01", "site 02", "site 03")
age <- c("19", "20", "22")
sex <- c("f", "m", "f")
gluc_post <- c(5.5, 4.1, 9.7)
gluc_res_post <- c("norm", "no", "cl")
ast_post <- c("30", "48", "31")
ast_res_post <- c(NA, "norm", "norm")

df <- data.frame(
  id, site, age, sex,
  gluc_post, gluc_res_post,
  ast_post, ast_res_post,
  stringsAsFactors = FALSE
)

refs <- system.file("labs_refer.xlsx", package = "dmtools")
obj_lab <- lab(refs, id, age, sex, "norm", "no")
```

```
obj_lab <- check(obj_lab, df)
choose_test(obj_lab, "mis")
```

`create_norm` *Estimating laboratory values*

Description

Estimating laboratory values

Usage

```
create_norm(vals, low, high, ds_norm, normal, abnormal, clsig)
```

Arguments

`vals`	A double vector. The laboratory values.
`low`	A double scalar. The minimum.
`high`	double scalar. The maximum.
`ds_norm`	An estimate of the laboratory values from the dataset.
`normal`	An option for the normal estimate, for example, "NORMAL".
`abnormal`	An option for the abnormal estimate, for example, "ABNORMAL".
`clsig`	An option for the clinical significant estimate, for example, "CLISIG".

Value

A vector with the auto estimate.

`create_spec` *For creating part of the specification*

Description

For creating part of the specification

Usage

```
create_spec(df_spec, all_colname, part_spec, is_pst)
```

Arguments

`df_spec`	A dataset, a type is a data frame.
`all_colname`	A character vector with all names in the dataset.
`part_spec`	A character scalar. Prefixes or postfixes.
`is_pst`	A logical scalar, default is TRUE. True is postfix, otherwise, prefix.

Value

A data frame. Part of the specification.

`date` *Create object date*

Description

Create object date

Usage

```
date(file, id, get_visit, get_date = dplyr::contains, str_date = "DAT")
```

Arguments

`file`	A character scalar. Path to the date's parameters in the excel table.
`id`	A column name of the subject id in the dataset, without quotes.
`get_visit`	A function, which select necessary visit or event e.g. dplyr::start_with, dplyr::contains.
`get_date`	A function, which select dates from necessary visit e.g. dplyr::matches, dplyr::contains, default: dplyr::contains.
`str_date`	A date's pattern in column names, default: "DAT".

Value

The object date.

Examples

```
obj_date <- date("dates.xlsx", id, dplyr::contains)
obj_date <- date("dates.xlsx", id, dplyr::contains, "uneq")
```

`dmtools` *dmtools: package to validate data*

Description

for checking the dataset from EDC in clinical trials

find_colnames *Find column names*

Description

Find column names

Usage

```
find_colnames(obj, dataset, row_file)
```

Arguments

obj	An object for check.
dataset	A dataset, a type is a data frame.
row_file	A row of the file.

Value

A data frame. Result of run_tests.

find_colnames.date *Find column names with dates*

Description

Find column names with dates

Usage

```
## S3 method for class 'date'
find_colnames(obj, dataset, row_file)
```

Arguments

obj	An object for validation.
dataset	A data frame. Class date.
row_file	A data frame. A data frame with analysis parameters.

Value

A data frame. Visit's dates.

find_colnames.default *Find column names*

Description

Find column names

Usage

```
## Default S3 method:
find_colnames(obj, dataset, row_file)
```

Arguments

obj	An object for validation.
dataset	A dataset, a type is a data frame.
row_file	A row of the file.

Value

A data frame. Result of run_tests.

get_result *Get the final result*

Description

Get the final result

Usage

```
get_result(obj, group_id)
```

Arguments

obj	An object. Can be all classes: short, lab, date.
group_id	A logical scalar, default is TRUE.True is grouped by id, otherwise, it isn't grouped.

Value

A data frame. The final result.

Examples

```
id <- c("01", "02", "03")
site <- c("site 01", "site 02", "site 03")
sex <- c("f", "m", "f")
preg_yn_e2 <- c("y", "y", "y")
preg_res_e2 <- c("neg", "neg", "neg")
preg_yn_e3 <- c("y", "y", "n")
preg_res_e3 <- c("neg", "pos", "unnes")

df <- data.frame(
  id, site, sex,
  preg_yn_e2, preg_res_e2,
  preg_yn_e3, preg_res_e3,
  stringsAsFactors = FALSE
)

preg <- system.file("preg.xlsx", package = "dmtools")
obj_short <- short(preg, id, "LBORRES", c("site", "sex"))

obj_short <- check(obj_short, df)
get_result(obj_short)
```

---

get_result.default *Get final result*

Description

Get final result

Usage

```
## Default S3 method:
get_result(obj, group_id = T)
```

Arguments

obj	An object. Can be all classes: short, lab, date.
group_id	A logical scalar, default is TRUE.True is grouped by id, otherwise, it isn't grouped.

Value

A data frame. The final result.

Examples

```
id <- c("01", "02", "03")
site <- c("site 01", "site 02", "site 03")
sex <- c("f", "m", "f")
preg_yn_e2 <- c("y", "y", "y")
preg_res_e2 <- c("neg", "neg", "neg")
preg_yn_e3 <- c("y", "y", "n")
preg_res_e3 <- c("neg", "pos", "unnes")

df <- data.frame(
  id, site, sex,
  preg_yn_e2, preg_res_e2,
  preg_yn_e3, preg_res_e3,
  stringsAsFactors = FALSE
)

preg <- system.file("preg.xlsx", package = "dmtools")
obj_short <- short(preg, id, "LBORRES", c("site", "sex"))

obj_short <- check(obj_short, df)
get_result(obj_short)
```

---

lab *Create object lab*

Description

Create object lab

Usage

```
lab(
  file,
  id,
  age,
  sex,
  normal,
  abnormal,
  is_post = T,
  clsig = NULL,
  site = NA,
  name_to_find = "LBNDIND"
)
```

Arguments

file	A character scalar. Path to the laboratory's reference in the excel table.
id	A column name of the subject id in the dataset, without quotes.

age	A column name of the subject age in the dataset, without quotes.
sex	A column name of the subject sex in the dataset, without quotes.
normal	A normal estimate, for example, "NORMAL".
abnormal	An abnormal estimate, for example, "ABNORMAL".
is_post	A logical scalar, default is TRUE. True is postfix, otherwise, prefix.
clsig	A clinical significant estimate, for example, "CLISIG".
site	A site number, default: NA.
name_to_find	A character scalar. For search prefixes or postfixes, default is "LBNDIND".

Value

The object lab.

Examples

```
obj_lab <- lab("lab_refer.xlsx", id, age, sex, 1, 2)
obj_lab <- lab("lab_refer.xlsx", id, age, sex, "NORMAL", "NOCLISIG", clsig = "CLISIG")
obj_lab <- lab("lab_refer.xlsx", id, age, sex, "norm", "no", FALSE)
```

---

list_parse *A list to a tibble.*

Description

A list to a tibble.

Usage

```
list_parse(to_tibble)
```

Arguments

to_tibble	A list with nested lists.

Value

A tibble.

Examples

```
temp_list <- list(list(a = 1, b = 3), list(a = 4, b = 5))
list_parse(temp_list)
```

meddra_auth *Get the token*

Description

Get the token

Usage

```
meddra_auth(target_url, meddra_id, api_key)
```

Arguments

target_url	The url for authenticate.
meddra_id	The user's meddra id.
api_key	The user's api key.

Value

A string scalar. The user's token.

Examples

```
## Not run:
meddra_auth(url, id, key)

## End(Not run)
```

meddra_post *Create the post query*

Description

Create the post query

Usage

```
meddra_post(target_url, json, token)
```

Arguments

target_url	The url for a post query.
json	A string scalar or a list. The json query.
token	The user's token.

Value

A list. The result of query.

Examples

```
## Not run:
meddra_post(url, json_body, token)

## End(Not run)
```

`rename_dataset` *For rename dataset*

Description

For rename dataset

Usage

```
rename_dataset(
  dataset,
  path_crfs,
  no_readable_name,
  readable_name,
  num_sheet = 1,
  extension = "*.xlsx",
  is_post = T
)
```

Arguments

`dataset`	A dataset, a type is a data frame.
`path_crfs`	A character scalar. Path to the specification files the in excel table.
`no_readable_name`	A character scalar. A column name of no_readable values.
`readable_name`	A character scalar. A column name of readable values.
`num_sheet`	An integer scalar, default is the first sheet. A position of a sheet in the excel document.
`extension`	A character scalar. A extension of files, default is *.xlsx.
`is_post`	A logical scalar, default is TRUE. True is postfix, otherwise, prefix.

Value

The list with two values: data - renamed dataset, spec - common specification. The common specification is data frame of two values: no_readable_var, readable_var.

Examples

```
id <- c("01", "02", "03")
age <- c("19", "20", "22")
sex <- c("f", "m", "f")
bio_date_post <- c("1991-03-23", "1991-03-16", "1991-03-16")
gluc_post <- c("5.5", "4.1", "9.7")
gluc_res_post <- c("norm", "no", "norm")


df <- data.frame(
  id, age, sex,
  bio_date_post,
  gluc_post, gluc_res_post,
  stringsAsFactors = FALSE
)

crfs <- system.file("forms", package = "dmtools")

result <- rename_dataset(df, crfs, "old_name", "new_name")
result[["data"]]
```

`run_tests` *Run tests*

Description

Run tests

Usage

```
run_tests(obj, dataset, row_file, part)
```

Arguments

`obj`	An object for check.
`dataset`	A data frame.
`row_file`	A data frame. A data frame with parameters.
`part`	A character scalar. Prefixes or postfixes.

Value

A data frame. The part of final result.

`run_tests.date` *Run test*

Description

Run test

Usage

```
## S3 method for class 'date'
run_tests(obj, dataset, row_file, date)
```

Arguments

`obj`	An object for validation.
`dataset`	A data frame. Class date.
`row_file`	A data frame. A data frame with analysis parameters.
`date`	A column name with dates.

Value

A data frame. Result of the date's validation.

`run_tests.lab` *Run tests*

Description

Run tests

Usage

```
## S3 method for class 'lab'
run_tests(obj, dataset, row_file, part)
```

Arguments

`obj`	An object. Class lab.
`dataset`	A data frame.
`row_file`	A data frame. A data frame with parameters.
`part`	A character scalar. Prefixes or postfixes.

Value

A data frame. The part of the final result.

run_tests.short *Run tests*

Description

Run tests

Usage

```
## S3 method for class 'short'
run_tests(obj, dataset, row_file, part)
```

Arguments

obj	An object. Class short.
dataset	A data frame.
row_file	A data frame. A data frame with parameters.
part	A character scalar. Prefixes or postfixes.

Value

A data frame. The part of the final result.

short *Create object short*

Description

Create object short

Usage

```
short(
  file,
  id,
  name_to_find,
  common_cols = NULL,
  extra = NULL,
  is_post = T,
  is_add_cols = F
)
```

Arguments

file	A character scalar. Path to the excel table.
id	A column name of the subject id in the dataset, without quotes.
name_to_find	A character scalar. For search prefixes or postfixes.
common_cols	A character vector. A column names in the dataset, which common for all events.
extra	A character scalar. For additional information.
is_post	A logical scalar, default is TRUE. True is postfix, otherwise, prefix.
is_add_cols	A logical scalar, default is FALSE. If necessary add columns.

Value

The object short.

Examples

```
obj_short <- short("preg.xlsx", id, "res", c("site", "sex"))
obj_short <- short("labs.xlsx", id, "name_labs", c("site"), "human_name")
```

---

test_sites *Test sites*

Description

Test sites

Usage

```
test_sites(objs, func)
```

Arguments

objs	A list of objects.
func	A function e.g. choose_test, get_result.

Value

A data frame. The dataset.

Examples

```
site <- c("site 01", "site 02")
id <- c("01", "02")
age <- c("19", "20")
sex <- c("f", "m")
gluc_post <- c("5.5", "4.1")
gluc_res_post <- c("norm", "no")
ast_post <- c("30", "48")
ast_res_post <- c(NA, "norm")

df <- data.frame(
  site, id, age, sex,
  gluc_post, gluc_res_post,
  ast_post, ast_res_post,
  stringsAsFactors = FALSE
)

refs_s01 <- system.file("labs_refer_s01.xlsx", package = "dmtools")
refs_s02 <- system.file("labs_refer_s02.xlsx", package = "dmtools")

s01_lab <- lab(refs_s01, id, age, sex, "norm", "no", site = "site 01")
s02_lab <- lab(refs_s02, id, age, sex, "norm", "no", site = "site 02")

labs <- list(s01_lab, s02_lab)
labs <- check_sites(labs, df, site)

test_sites(labs, func = function(lab) choose_test(lab, "mis"))
```

---

to_dbl *Cast to double type*

Description

Cast to double type

Usage

```
to_dbl(vals)
```

Arguments

vals	A character or double vector.

Value

A double vector.

Index