Package 'DQAstats'

April 4, 2024

Title Core Functions for Data Quality Assessment

Version 0.3.4

Date 2024-04-04

Description Perform data quality assessment ('DQA') of electronic health records ('EHR'). Publication: Kapsner et al. (2021) <doi:10.1055/s-0041-1733847>.

License GPL-3

URL https://github.com/miracum/dqa-dqastats

BugReports https://github.com/miracum/dqa-dqastats/issues

Imports data.table, DIZtools (>= 0.0.8), DIZutils (>= 0.1.2), future, future.apply, jsonlite, kableExtra, knitr, magrittr, parsedate, rmarkdown, stats, tinytex, utils

Suggests DT, lintr, remotes, testthat

VignetteBuilder knitr

Config/testthat/edition 3

Config/testthat/parallel false

Encoding UTF-8

NeedsCompilation no

Author Lorenz A. Kapsner [cre, aut] (<https://orcid.org/0000-0003-1866-860X>), Jonathan M. Mang [aut] (<https://orcid.org/0000-0003-0518-4710>), MIRACUM - Medical Informatics in Research and Care in University Medicine [fnd], Universitätsklinikum Erlangen [cph]

Maintainer Lorenz A. Kapsner <lorenz.kapsner@gmail.com>

Repository CRAN

Date/Publication 2024-04-04 08:33:00 UTC

R topics documented:

`apply_time_restriciton` *Time filtering of data.table or sql-strings.*

Description

Internal function to filter the input data (or SQL) depending on provided time information. Sensitive to SQL dialects.

Usage

```
apply_time_restriciton(
  data,
  key,
  lower_limit,
  upper_limit,
  system_name = NULL,
```

```
  system_type,
  mdr,
  logfile_dir = NULL,
  db_con = NULL,
  sql_create_view_all = list(),
  verify_on_db = TRUE
)
```

Arguments

data	If system_type is a database, the sql-string goes here. If system_type is 'csv', the data.table of this csv goes here. Sensitive to SQL dialects.
key	The key from the mdr.
lower_limit	The posixct timestamp of the lower filtering boundary.
upper_limit	The posixct timestamp of the upper filtering boundary.
system_name	(Optional for non-database-changes) 'i2b2'/'p21csv'/'omop'/...
system_type	'postgres'/'oracle'/'csv'
mdr	(Optional for non-database-changes) The internal MDR (get it from rv$mdr)
logfile_dir	(Optional) The directory to store the logfile in. Defaults to NULL.
db_con	(Optional for non-database-changes) The connection to the database. Used to create the views we need later to apply the SQLs to.
sql_create_view_all	(Optional, list). A list containing the SQLs to create all Views for the time-filtering. This is needed for the printing-friendly SQL including this view-creating SQLs and the actual data-extracting SQL query.
verify_on_db	A boolean. If the view should be verified on the database (default: TRUE).

Value

If system_type is a database, a list with the new sql-string containing the temporal filtering will be returned under $sql ('order by' parts will be removed) and a printable sql containing the commands to create the view needed to run the sql under $sql_extended. If system_type is 'csv', the filtered data.table will be returned.

---

atemp_plausi_results *atemp_plausi_results helper function*

Description

Internal function to generate the results of the 'Atemporal Plausibility' checks.

Usage

```
atemp_plausi_results(rv, atemp_vars, mdr, headless = FALSE)
```

Arguments

rv	A list object. Internal list simulating Shiny's 'reactive values'.
atemp_vars	These are the atemporal variables
mdr	A data.table object containing the MDR.
headless	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).

Value

A list with one entry for each atemporal plausibility check containing the results. Each entry contains the following (nested) list items:

description A nested list with the description of the plausibility check for the source data system and the target data system.

counts A nested list with the frequency count results for the source data system and the target data system.

statistics A nested list with the plausibility check results for the source data system and the target data system.

Examples

```
# runtime ~ 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
```

```
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
```

```
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
  gc()
}

# calculate descriptive results
rv$results_descriptive <- descriptive_results(
  rv = rv,
  headless = rv$headless
)

# calculate atemporal plausibilites
atemp_plausi_results(
  rv = rv,
  atemp_vars = rv$data_plausibility$atemporal,
  mdr = rv$mdr,
  headless = rv$headless
)
```

`check_date_restriction_requirements` *Checking the mdr integrity for time restrictions*

Description

Internal function to check if for every input table there is one single (or empty) column where to apply the time restriction to. If the input is valid, it will just print a success-message, if the data is invalid, the function will call `stop()`.

Usage

```
check_date_restriction_requirements(
  mdr,
  system_names,
  logfile_dir,
  headless = TRUE,
  enable_stop = TRUE
)
```

Arguments

`mdr`	The mdr as data.table
`system_names`	(String) The name of the systems (source and target) to check for possible date restriction in the mdr.

logfile_dir	The absolute path to folder where the logfile will be stored default(tempdir()).
headless	(Boolean) Is this a console application? Otherwise (if headless = FALSE) there is a GUI and there will be GUI-feedback.
enable_stop	(Boolean, default = TRUE) If true (default) this function will call stop() in case of the check fails. If enable_stop = FALSE it will return TRUE if the check was successful and FALSE if the check failed. Use enable_stop = FALSE to avoid the need of a try/catch block around this function.

Value

TRUE/FALSE: TRUE if the check was successful and the given systems can be time filtered, FALSE if something went wrong and no time filtering is possible.

A boolean to indicate if the date restriction requirements are met (TRUE) or not (FALSE).

Examples

```
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename = mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

DIZtools::cleanup_old_logfile(logfile_dir = tempdir())

check_date_restriction_requirements(
  mdr = mdr,
  system_names = c(source_system_name, target_system_name),
  logfile_dir = tempdir(),
  headless = TRUE,
  enable_stop = TRUE
)
```

---

completeness *completeness helper function*

Description

Internal function to perform missing analysis.

Usage

```
completeness(results, headless = FALSE, logfile_dir)
```

Arguments

results	A list object. The list should contain the results of either 'rv$results_descriptive' or 'rv$results_plausibility_atemporal'.
headless	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).
logfile_dir	The absolute path to folder where the logfile will be stored default(`tempdir()`).

Value

A data.table with the absolute and relative counts of missing values (results of the completeness checks) for each dataelement for the source data system and the target data system.

Examples

```
# runtime ~ 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
```

```
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
```

```
  gc()
}

# calculate descriptive results
rv$results_descriptive <- descriptive_results(
  rv = rv,
  headless = rv$headless
)
completeness(
  results = rv$results_descriptive,
  headless = rv$headless,
  logfile_dir = rv$log$logfile_dir
)
```

`create_helper_vars` *create_helper_vars helper function*

Description

Internal function to create necessary variables from the meta data repository (MDR).

Usage

```
create_helper_vars(mdr, source_db, target_db)
```

Arguments

`mdr`	A data.table object containing the MDR.
`source_db`	A character string. The name of the source database. This string must be conform with the corresponding config section in the config.yml-file.
`target_db`	A character string. The name of the target database. This string must be conform with the corresponding config section in the config.yml-file.

Value

A list with results from the analysis of the metadata repository (MDR) with the following items:

keys_source A character vector with the different values of the 'key' field from the MDR for the source data system.

keys_target A character vector with the different values of the 'key' field from the MDR for the target data system.

dqa_assessment A data.table with a subset of the MDR for the dataelement entries with the field 'dqa_assessment' = 1.

variable_list A mapping list from MDR variable names (MDR field 'designation') to DQA tool internal variable names (MDR field 'variable_name').

pl A nested list with items regarding the plausibility checks

- **atemp_vars** A data.table with a subset of the MDR with dataelements that are associated with atemporal plausibility checks.
- **uniq_vars** A data.table with a subset of the MDR with dataelements that are associated with uniqueness plausibility checks.
- **atemp_helper_vars** A character vector with further dataelements that are required to perform the atemporal plausibility checks.
- **atemp_possible** A boolean to indicate if all dataelements required to perform the atemporal plausibility checks are available in the dataset.
- **uniq_helper_vars** A character vector with further dataelements that are required to perform the uniqueness plausibility checks.
- **uniq_possible** A boolean to indicate if all dataelements required to perform the uniqueness plausibility checks are available in the dataset.

Examples

```
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename = mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

create_helper_vars(
  mdr = mdr,
  source_db = source_system_name,
  target_db = target_system_name
)
```

---

create_markdown *create_markdown helper function*

Description

Internal function to generate the final PDF report.

Usage

```
create_markdown(rv = rv, utils_path, outdir = tempdir(), headless = FALSE)
```

Arguments

rv	A list object. Internal list simulating Shiny's 'reactive values'.
utils_path	A character string. The path to the utils-folder, containing the required app utilities like the MDR and the settings folder.
outdir	A character string. The directory to store the resulting PDF document. Default: tempdir.
headless	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).

Value

No return value. This function renders the PDF markdown report with the data quality assessment results and saves it to outdir.

Examples

```
# runtime > 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
```

```
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
  gc()
```

```
}

# calculate descriptive results
rv$results_descriptive <- descriptive_results(
  rv = rv,
  headless = rv$headless
)

# calculate unique plausibilites
rv$results_plausibility_unique <- uniq_plausi_results(
  rv = rv,
  uniq_vars = rv$pl$uniq_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

create_markdown(
  rv = rv,
  utils_path = rv$utilspath,
  outdir = output_dir,
  headless = rv$headless
)
```

`data_loading` *data_loading helper function*

Description

Internal function to load the source and target data

Usage

```
data_loading(rv, system, keys_to_test)
```

Arguments

`rv`	The complete reactive-value dataset
`system`	The part of the rv-list which should be loaded (e.g. rv$source or rv$target)
`keys_to_test`	A vector containing the names (keys) of the variables to test.

Value

A list with the fields '$outdata' and if testing an SQL-based database also '$sql_statements'.

Examples

```
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename = mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
```

```
rv$restricting_date$use_it <- FALSE

data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
```

`descriptive_results` *descriptive_results helper function*

Description

Internal function to generate the descriptive results.

Usage

```
descriptive_results(rv, headless = FALSE)
```

Arguments

`rv`	A list object. Internal list simulating Shiny's 'reactive values'.
`headless`	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).

Value

A list with one entry for each dataelement containing the results of the descriptive results. Each entry contains the following (nested) list items:

description A nested list with the description of the dataelement for the source data system and the target data system.

counts A nested list with the frequency count results for the source data system and the target data system.

statistics A nested list with the descriptive results for the source data system and the target data system stored as data.table objects.

Examples

```
# runtime ~ 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
```

```
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata
```

```
# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
  gc()
}

# calculate descriptive results
descriptive_results(
  rv = rv,
  headless = rv$headless
)
```

---

dqa *Perform Data Quality Assessment of Electronic Health Records.*

Description

This function performs a data quality assessment (DQA) of electronic health records (EHR).#'

Usage

```
dqa(
  source_system_name,
  target_system_name,
  utils_path,
  mdr_filename = "mdr.csv",
```

```
  output_dir = paste0(tempdir(), "/output/"),
  logfile_dir = tempdir(),
  parallel = FALSE,
  ncores = 2,
  restricting_date_start = NULL,
  restricting_date_end = NULL,
  restricting_date_format = NULL
)
```

Arguments

`source_system_name`
: A character string. The name of the source-system, e.g. "P21" or "i2b2". This name must be identical and unique to one entry in the settings-yml file.

`target_system_name`
: Optional. A character string or null. The name of the target-system, e.g. "P21" or "i2b2". This name must be identical and unique to one entry in the config-yml file or null. If the argument is empty, the source will be processed as standalone on its own.

`utils_path`
: A character string. The path to the utils-folder, containing the required app utilities like the MDR and the settings folder.

`mdr_filename`
: A character string. The filename of the MDR e.g. "mdr_example_data.csv".

`output_dir`
: The path to the output folder where all the results will be stored (default: `paste0(tempdir(), "/output/")`).

`logfile_dir`
: The absolute path to folder where the logfile will be stored default(`tempdir()`).

`parallel`
: A boolean. If TRUE, initializing a `future::plan()` for running the code (default: FALSE).

`ncores`
: A integer. The number of cores to use. Caution: you would probably like to choose a low number when operating on large datasets. Default: 2.

`restricting_date_start`
: The date as the lower limit against which the data to be analyzed will be filtered. Your input must be able to be recognized as a date by `parsedate::parse_date("2021-02-25")`. Keep in mind: If you supply a date without a time here, the time will automatically be set to 00:00.

`restricting_date_end`
: The date as the lower limit against which the data to be analyzed will be filtered. Your input must be able to be recognized as a date by `parsedate::parse_date("2021-02-25")` Keep in mind: If you supply a date without a time here, the time will automatically be set to 00:00. This means, the end DAY you provide here won't be included: '2021-12-31' will become '2021-12-31 00:00:00'. If you want to include this day, you need to supply also a time '2021-12-31 23:59:59' or just use the next day without a time: '2022-01-01'.

`restricting_date_format`
: The format in which the input data is stored. See `?strptime` for possible parameters. Currently not implemented! So there is no effect if you pass a format here.

Value

This function is a wrapper around all helper functions in DQAstats to perform the data quality assessment. The results are summarized in a PDF report which is saved to outdir. The return value of this function is a nested list that contains all results as R objects.

Examples

```
# runtime > 5 sec.
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = system.file(
  "demo_data",
  package = "DQAstats")
)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = system.file(
  "demo_data",
  package = "DQAstats")
)

# Set path to utilities folder where to find the mdr and template files:
utils_path <- system.file(
  "demo_data/utilities",
  package = "DQAstats"
)

# Execute the DQA and generate a PDF report:
results <- DQAstats::dqa(
  source_system_name = "exampleCSV_source",
  target_system_name = "exampleCSV_target",
  utils_path = utils_path,
  mdr_filename = "mdr_example_data.csv",
  output_dir = paste0(tempdir(), "/output/"),
  parallel = FALSE
)
```

---

etl_checks *etl_checks helper function*

Description

Internal function to perform etl conformance checks.

Usage

```
etl_checks(results)
```

Arguments

results	A list object. The list should contain the results 'rv$results_descriptive'.

Value

A data.table with the automated comparison of the counts of valid, missing, and distinct values between the source data system and the target data system.

Examples

```
# runtime ~ 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
```

```
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
  gc()
}

# calculate descriptive results
rv$results_descriptive <- descriptive_results(
  rv = rv,
  headless = rv$headless
)

etl_checks(results = rv$results_descriptive)
```

`export_aggregated` *Export results to csv/zip file.*

Description

This function exports aggregated results in csv files that are added to a zip archive.

Usage

```
export_aggregated(output_dir, rv)
```

Arguments

`output_dir`	The path to the output folder where all the results will be stored (default: `paste0(tempdir(), "/output/")`).
`rv`	A list object. Internal list simulating Shiny's 'reactive values'.

Value

No return value. This function writes the aggregated results, namely the conformace results overview table, the plausibility check results overview, the completeness results overview and a combined version (named 'all_results.csv') to csv files. These files are saved in `{output_dir}/export`.

Examples

```
# runtime > 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
```

```
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
)
```

```
# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
  gc()
}

# calculate descriptive results
rv$results_descriptive <- descriptive_results(
  rv = rv,
  headless = rv$headless
)

# completeness
rv$completeness <- completeness(results = rv$results_descriptive,
                                headless = rv$headless,
                                logfile_dir = rv$log$logfile_dir)

rv$datamap <- generate_datamap(
  results = rv$results_descriptive,
  db = rv$target$system_name,
  mdr = rv$mdr,
  rv = rv,
  headless = rv$headless
)

# checks$value_conformance
rv$checks$value_conformance <-
  value_conformance_checks(results = rv$conformance$value_conformance)

# checks$etl
rv$checks$etl <- etl_checks(results = rv$results_descriptive)

output_dir <- tempdir()
export_aggregated(
  output_dir = output_dir,
  rv = rv
)
```

---

format_value_conformance_results

format_value_conformance_results helper function

Description

Internal function to format the value conformance results

Usage

```
format_value_conformance_results(results, desc_out, source)
```

Arguments

results	A list containing the value conformance results for one data element.
desc_out	A list containing the descriptive results for the same data element.
source	A character: either source_data or target_data to indicate, which results to render.

Value

The function returns a list with the formatted value conformance results for one data element.

Examples

```
# runtime ~ 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE
```

```
# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
```

```
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
  gc()
}

# calculate descriptive results
rv$results_descriptive <- descriptive_results(
  rv = rv,
  headless = rv$headless
)

# calculate atemporal plausibilites
rv$results_plausibility_atemporal <- atemp_plausi_results(
  rv = rv,
  atemp_vars = rv$data_plausibility$atemporal,
  mdr = rv$mdr,
  headless = rv$headless
)

# calculate unique plausibilites
rv$results_plausibility_unique <- uniq_plausi_results(
  rv = rv,
  uniq_vars = rv$pl$uniq_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

rv$conformance$value_conformance <- value_conformance(
  rv = rv,
  scope = "descriptive",
  results = rv$results_descriptive,
  headless = rv$headless,
  logfile_dir = rv$log$logfile_dir
)

# format the results (wrap functioncall into `sapply` to get results for all
# available data elements):
value_conformance_formatted <- sapply(
  X = names(rv$results_descriptive),
  FUN = function(i) {
    desc_out <- rv$results_descriptive[[i]]$description

    if (!is.null(rv$conformance$value_conformance[[i]])) {
      format_value_conformance_results(
        results = rv$conformance$value_conformance[[i]],
        desc_out = desc_out,
        source = "source_data"
      )
    }
  }
)
```

generate_datamap *generate_datamap helper function*

Description

Internal function to generate the dashboard data maps

Usage

```
generate_datamap(results, mdr, db, rv, headless = FALSE)
```

Arguments

results	A list object. The list should contain the results 'rv$results_descriptive'.
mdr	A data.table object containing the MDR.
db	A character string. The name of the corresponding database.
rv	A list object. Internal list simulating Shiny's 'reactive values'.
headless	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).

Value

A data.table with the results of the datamap.

Examples

```
# runtime > 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
```

```
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
```

```
)

# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
  gc()
}

# calculate descriptive results
rv$results_descriptive <- descriptive_results(
  rv = rv,
  headless = rv$headless
)

generate_datamap(
  results = rv$results_descriptive,
  db = rv$target$system_name,
  mdr = rv$mdr,
  rv = rv,
  headless = rv$headless
)
```

`get_atemp_plausis` *get_atemp_plausis helper function*

Description

Internal function to generate raw data for the 'Atemporal Plausibility' checks.

Usage

```
get_atemp_plausis(rv, atemp_vars, mdr, headless = FALSE)
```

Arguments

`rv`	A list object. Internal list simulating Shiny's 'reactive values'.
`atemp_vars`	A data.table object. The object is created by `create_helper_vars` from the data represented in the metadata repository.
`mdr`	A data.table object containing the MDR.
`headless`	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).

Value

A list with one entry for each atemporal plausibility check containing the raw results. Each entry contains the following (nested) list items:

source_data A nested list with the raw plausibility check results for the source data system.

target_data A nested list with the raw plausibility check results for the target data system.

Examples

```
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename = mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
```

```
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
)
```

`get_restricting_date_info` *Get a formatted string containing start and end time of the date restriction applied to the data.*

Description

See title.

Usage

```
get_restricting_date_info(
  restricting_date,
  lang = "en",
  date = TRUE,
  time = TRUE
)
```

Arguments

restricting_date	The list applied from rv$restricting_date
lang	Language of the result. "de"/"en" (en = default). If language is not yet implemented, "en" is used.
date	Should the date be included in the result string?
time	Should the time be included in the result string?

Value

String containing start and end date obtained from the list of `restricting_date`.

---

is_latex_installed *Checks if there is a LaTeX installation available*

Description

Internal function to determine if a LaTeX installation is available. Used before creating/knitr-ing the PDF report.

Usage

```
is_latex_installed(logfile_dir = NULL, headless = TRUE)
```

Arguments

logfile_dir	The absolute path to folder where the logfile will be stored default(`tempdir()`).
headless	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).

Value

TRUE if there is a LaTeX installation, FALSE if not.

Examples

```
is_latex_installed()
```

load_csv *load_csv helper function*

Description

Internal function to load the data from CSV files.

Usage

```
load_csv(rv, keys_to_test, headless = FALSE, system)
```

Arguments

rv	A list object. Internal list simulating Shiny's 'reactive values'.
keys_to_test	A vector containing the names (keys) of the variables to test.
headless	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).
system	The system object rv$system

Value

A list with data.tables for each unique CSV file as defined in the 'source_system_table' field of the MDR.

load_database *load_database helper function*

Description

Internal function to load the data from SQL databases.

Usage

```
load_database(
  rv,
  sql_statements,
  db_con,
  keys_to_test,
  db_name,
  headless = FALSE,
  db_type
)
```

Arguments

rv	A list object. Internal list simulating Shiny's 'reactive values'.
sql_statements	The SQL-Statement-object
db_con	The connection-socket
keys_to_test	A vector containing the names (keys) of the variables to test.
db_name	The database name
headless	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).
db_type	The database type (postgres/oracle)

Value

A list with a data.table for each data element as defined in the in the MDR.

---

load_sqls *load_sqls helper function*

Description

Internal function to load the SQL statements.

Usage

```
load_sqls(utils_path, db)
```

Arguments

utils_path	A character string. The path to the utils-folder, containing the required app utilities like the MDR and the settings folder.
db	A character string. The name of the corresponding database.

---

parallel *parallel helper function*

Description

Internal function to initialize the parallel backend.

Usage

```
parallel(parallel, logfile_dir, ncores)
```

Arguments

parallel	A boolean. If TRUE, initializing a future::plan() for running the code (default: FALSE).
logfile_dir	The absolute path to folder where the logfile will be stored default(tempdir()).
ncores	A integer. The number of cores to use. Caution: you would probably like to choose a low number when operating on large datasets. Default: 2.

Value

No return value. Depending on the specified arguments, this function enables a parallel backend for faster computations.

Examples

```
parallel(parallel = FALSE, logfile_dir = tempdir(), ncores = 1)
```

---

read_mdr *read_mdr helper function*

Description

Internal function to read the meta data repository (MDR).

Usage

```
read_mdr(utils_path = NULL, mdr_filename = "mdr.csv")
```

Arguments

utils_path	A character string. The path to the utils-folder, containing the required app utilities like the MDR and the settings folder.
mdr_filename	A character string. The filename of your meta data repository (default: 'mdr.csv').

Value

A data.table containing the metadata repository which is imported from the CSV file provided with {utils_path}/MDR/{mdr_filename}.

Examples

```
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename = mdr_filename
)
```

`reduce_cat` *reduce_cat helper function*

Description

Internal function to reduce categorical variables to a maximum of values to be displayed.

Usage

```
reduce_cat(data, levellimit = 25)
```

Arguments

`data`	A list object. The object `rv$results_descriptive`.
`levellimit`	An integer value. The number of maximum values to be displayed (default: 25).

Value

A data.table with the data quality assessment results for categorical dataelements that are reduced to the maximum number of levels specified with `levellimit`.

Examples

```
# runtime ~ 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"
```

```
rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
```

```
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
  gc()
}

# calculate descriptive results
rv$results_descriptive <- descriptive_results(
  rv = rv,
  headless = rv$headless
)

reduce_cat(
  data = rv$results_descriptive,
  levellimit = 25
)
```

`test_csv` *test_csv helper function*

Description

Internal function to test and get the database connection of the source database.

Usage

```
test_csv(settings, source_db, mdr, headless = FALSE, logfile_dir)
```

Arguments

`settings`	A list object containing the database settings.

source_db	A character string. The name of the source database. This string must be conform with the corresponding config section in the config.yml-file.
mdr	A data.table object containing the MDR.
headless	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).
logfile_dir	The absolute path to folder where the logfile will be stored default(tempdir()).

Value

A boolean indicating if the CSV files specified in the metadata repository are found in the specified locations.

Examples

```
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename = mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

DIZtools::cleanup_old_logfile(logfile_dir = tempdir())

settings <- DIZutils::get_config_env(
  system_name = source_system_name,
  logfile_dir = tempdir(),
  headless = TRUE
)

test_csv(
  settings = settings,
  source_db = source_system_name,
  mdr = mdr,
  headless = TRUE,
  logfile_dir = tempdir()
)
```

---

uniq_plausi_results *uniq_plausi_results helper function*

Description

Internal function to generate the results of the 'Uniqueness Plausibility' checks.

Usage

```
uniq_plausi_results(rv, uniq_vars, mdr, headless = FALSE)
```

Arguments

rv	A list object. Internal list simulating Shiny's 'reactive values'.
uniq_vars	A data.table object. The object is created by create_helper_vars from the data represented in the metadata repository.
mdr	A data.table object containing the MDR.
headless	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).

Value

A list with one entry for each uniqueness plausibility check containing the results. Each entry contains the following (nested) list items:

description A character with the description of the plausibility check.

source_data A nested list with the uniqueness plausibility check results for the source data system with the values 'message' and 'error'.

target_data A nested list with the uniqueness plausibility check results for the target data system with the values 'message' and 'error'.

Examples

```
# runtime > 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
```

```
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
```

```
  mdr = rv$mdr,
  headless = rv$headless
)

# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
  gc()
}

# calculate descriptive results
rv$results_descriptive <- descriptive_results(
  rv = rv,
  headless = rv$headless
)

# calculate unique plausibilites
uniq_plausi_results(
  rv = rv,
  uniq_vars = rv$pl$uniq_vars,
  mdr = rv$mdr,
  headless = rv$headless
)
```

`value_conformance` *value_conformance helper function*

Description

Internal function to perform value conformance checks.

Usage

```
value_conformance(rv, results, scope, headless = FALSE, logfile_dir)
```

Arguments

`rv`	A list object. Internal list simulating Shiny's 'reactive values'.
`results`	A list object. The list should contain the results of either 'rv$results_descriptive' or 'rv$results_plausibility_atemporal'.
`scope`	A character. Either "plausibility" or "descriptive".

headless	A boolean (default: FALSE). Indicating, if the function is run only in the console (headless = TRUE) or on a GUI frontend (headless = FALSE).
logfile_dir	The absolute path to folder where the logfile will be stored default(tempdir()).

Value

A list with one entry for each dataelement containing the raw results of the value conformance checks. Each entry contains the following (nested) list items:

source_data A nested list with the raw value conformance check results for the source data system.

target_data A nested list with the raw value conformance check results for the target data system.

Examples

```
# runtime ~ 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
```

```
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")

# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
  gc()
}

# calculate descriptive results
rv$results_descriptive <- descriptive_results(
```

```
  rv = rv,
  headless = rv$headless
)

# calculate atemporal plausibilites
rv$results_plausibility_atemporal <- atemp_plausi_results(
  rv = rv,
  atemp_vars = rv$data_plausibility$atemporal,
  mdr = rv$mdr,
  headless = rv$headless
)

# calculate unique plausibilites
rv$results_plausibility_unique <- uniq_plausi_results(
  rv = rv,
  uniq_vars = rv$pl$uniq_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

value_conformance(
  rv = rv,
  scope = "descriptive",
  results = rv$results_descriptive,
  headless = rv$headless,
  logfile_dir = rv$log$logfile_dir
)
```

`value_conformance_checks` *value_conformance_checks helper function*

Description

Internal function to perform value conformance checks.

Usage

```
value_conformance_checks(results)
```

Arguments

`results`	A list object. The list should contain the results of the function `value_conformance`.

Value

A data.table with the results of the automated comparison of the value conformance check results between the source data system and the target data system.

Examples

```
# runtime ~ 5 sec.
utils_path <- system.file(
  "demo_data/utilities/",
  package = "DQAstats"
)
mdr_filename <- "mdr_example_data.csv"
rv <- list()
rv$mdr <- read_mdr(
  utils_path = utils_path,
  mdr_filename <- mdr_filename
)

source_system_name <- "exampleCSV_source"
target_system_name <- "exampleCSV_target"

rv <- c(rv, create_helper_vars(
  mdr = rv$mdr,
  source_db = source_system_name,
  target_db = target_system_name
))
# save source/target vars
rv$source$system_name <- source_system_name
rv$target$system_name <- target_system_name
rv$source$system_type <- "csv"
rv$target$system_type <- "csv"

rv$log$logfile_dir <- tempdir()

# set headless (without GUI, progressbars, etc.)
rv$headless <- TRUE

# set configs
demo_files <- system.file("demo_data", package = "DQAstats")
Sys.setenv("EXAMPLECSV_SOURCE_PATH" = demo_files)
Sys.setenv("EXAMPLECSV_TARGET_PATH" = demo_files)

# get configs
rv$source$settings <- DIZutils::get_config_env(
  system_name = rv$source$system_name,
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)
rv$target$settings <- DIZutils::get_config_env(
  system_name = tolower(rv$target$system_name),
  logfile_dir = rv$log$logfile_dir,
  headless = rv$headless
)

# set start_time (e.g. when clicking the 'Load Data'-button in shiny
rv$start_time <- format(Sys.time(), usetz = TRUE, tz = "CET")
```

```
# define restricting date
rv$restricting_date$use_it <- FALSE

# load source data
tempdat <- data_loading(
  rv = rv,
  system = rv$source,
  keys_to_test = rv$keys_source
)
rv$data_source <- tempdat$outdata

# load target data
tempdat <- data_loading(
  rv = rv,
  system = rv$target,
  keys_to_test = rv$keys_target
)
rv$data_target <- tempdat$outdata

rv$data_plausibility$atemporal <- get_atemp_plausis(
  rv = rv,
  atemp_vars = rv$pl$atemp_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

# add the plausibility raw data to data_target and data_source
for (i in names(rv$data_plausibility$atemporal)) {
  for (k in c("source_data", "target_data")) {
    w <- gsub("_data", "", k)
    raw_data <- paste0("data_", w)
    rv[[raw_data]][[i]] <-
      rv$data_plausibility$atemporal[[i]][[k]][[raw_data]]
    rv$data_plausibility$atemporal[[i]][[k]][[raw_data]] <- NULL
  }
  gc()
}

# calculate descriptive results
rv$results_descriptive <- descriptive_results(
  rv = rv,
  headless = rv$headless
)

# calculate atemporal plausibilites
rv$results_plausibility_atemporal <- atemp_plausi_results(
  rv = rv,
  atemp_vars = rv$data_plausibility$atemporal,
  mdr = rv$mdr,
  headless = rv$headless
)

# calculate unique plausibilites
```

```
rv$results_plausibility_unique <- uniq_plausi_results(
  rv = rv,
  uniq_vars = rv$pl$uniq_vars,
  mdr = rv$mdr,
  headless = rv$headless
)

rv$conformance$value_conformance <- value_conformance(
  rv = rv,
  scope = "descriptive",
  results = rv$results_descriptive,
  headless = rv$headless,
  logfile_dir = rv$log$logfile_dir
)

value_conformance_checks(results = rv$conformance$value_conformance)
```

Index