

# Package ‘dataquieR’

May 8, 2026

**Title** Data Quality in Epidemiological Research

**Version** 2.8.7

**Description** Data quality assessments guided by a 'data quality framework introduced by Schmidt and colleagues, 2021' <[doi:10.1186/s12874-021-01252-7](https://doi.org/10.1186/s12874-021-01252-7)> target the data quality dimensions integrity, completeness, consistency, and accuracy. The scope of applicable functions rests on the availability of extensive metadata which can be provided in spreadsheet tables. Either standardized (e.g. as 'html5' reports) or individually tailored reports can be generated. For an introduction into the specification of corresponding metadata, please refer to the 'package website' <[https://dataquality.qihs.uni-greifswald.de/VIN\\_Annotation\\_of\\_Metadata.html](https://dataquality.qihs.uni-greifswald.de/VIN_Annotation_of_Metadata.html)>.

**License** BSD\_2\_clause + file LICENSE

**URL** <https://dataquality.qihs.uni-greifswald.de/>

**BugReports** <https://gitlab.com/libreumg/dataquieR/-/issues>

**Depends** R (>= 3.6.0)

**Imports** dplyr (>= 1.0.2), emmeans, ggplot2 (>= 3.5.0), lme4, lubridate, MASS, MultinomialCI, parallelMap, patchwork (>= 1.3.0), R.devices, rlang, robustbase, qmrparser, utils, rio, readr, scales, withr, lifecycle, units, methods, hms

**Suggests** S7 (>= 0.2.1), cowplot, grid, openxlsx2, grDevices, jsonlite, cli, whoami, DT (>= 0.23), htmltools, knitr, markdown, parallel, parallelly, rmarkdown, rstudioapi, testthat (>= 3.1.9), tibble, vdiff, pkgload, Rdpack, callr, colorspace, plotly (>= 4.11.0), htmlwidgets, future, processx, R6, shiny, xml2, mgcv, rvest, textutils, dbx, grImport2, rsvg, stringdist, rankICC, nnet, ordinal, storr, reticulate, stringi, lobstr, visNetwork

**VignetteBuilder** knitr

**Encoding** UTF-8

**KeepSource** FALSE

**Language** en-US

**RoxygenNote** 7.3.3

**Config/testthat/parallel** true

**Config/testthat/edition** 3

**Config/testthat/start-first** dq\_report\_by\_sm, dq\_report2,  
 dq\_report\_by\_arguments, dq\_report\_by\_pipesymbol\_list,  
 dq\_report\_by\_s, dq\_report\_by\_m, util\_handle\_complex\_data\_types,  
 int\_encoding\_errors, plots, acc\_loess, com\_item\_missingness,  
 dq\_report\_by\_na, dq\_report\_by\_directories,  
 con\_limit\_deviations, con\_contradictions\_recap,  
 com\_segment\_missingness, util\_correct\_variable\_use

**BuildManual** TRUE

**NeedsCompilation** no

**Author** University Medicine Greifswald [cph],

Elisa Kasbohm [aut] (ORCID: <<https://orcid.org/0000-0001-5261-538X>>),

Elena Salogni [aut] (ORCID: <<https://orcid.org/0009-0007-3767-7145>>),

Joany Marino [aut] (ORCID: <<https://orcid.org/0000-0002-4657-3758>>),

Adrian Richter [aut] (ORCID: <<https://orcid.org/0000-0002-3372-2021>>),

Carsten Oliver Schmidt [aut] (ORCID:

<<https://orcid.org/0000-0001-5266-9396>>),

Stephan Struckmann [aut, cre] (ORCID:

<<https://orcid.org/0000-0002-8565-7962>>),

German Research Foundation (DFG SCHM 2744/3-1, SCHM 2744/9-1, SCHM  
 2744/3-4) [fnd],

National Research Data Infrastructure for Personal Health Data: (NFDI  
 13/1) [fnd],

European Union's Horizon 2020 programme (euCanSHare, grant agreement  
 No. 825903) [fnd]

**Maintainer** Stephan Struckmann <[stephan.struckmann@uni-greifswald.de](mailto:stephan.struckmann@uni-greifswald.de)>

**Repository** CRAN

**Date/Publication** 2026-01-08 09:20:09 UTC

## Contents

*.numeric_with_unit . . . . .	9
+numeric_with_unit . . . . .	9
-numeric_with_unit . . . . .	10
/.numeric_with_unit . . . . .	10
acc_cat_distributions . . . . .	11
acc_distributions . . . . .	12
acc_distributions_ecdf . . . . .	14
acc_distributions_loc . . . . .	15
acc_distributions_only . . . . .	17
acc_distributions_prop . . . . .	18
acc_end_digits . . . . .	20

acc_loess . . . . .	21
acc_mahalanobis . . . . .	24
acc_margins . . . . .	25
acc_multivariate_outlier . . . . .	28
acc_robust_univariate_outlier . . . . .	30
acc_shape_or_scale . . . . .	32
acc_univariate_outlier . . . . .	34
acc_varcomp . . . . .	36
as.character.interval . . . . .	38
as.data.frame.dataquieR_resultset . . . . .	39
as.list.dataquieR_resultset . . . . .	39
as.list.dataquieR_resultset2 . . . . .	40
ASSOCIATION_DIRECTION . . . . .	40
ASSOCIATION_FORM . . . . .	41
ASSOCIATION_METRIC . . . . .	41
ASSOCIATION_RANGE . . . . .	42
CHECK_ID . . . . .	42
CHECK_LABEL . . . . .	43
check_table . . . . .	44
CODE_CLASSES . . . . .	44
CODE_LIST_TABLE . . . . .	44
CODE_ORDER . . . . .	45
COMPUTATION_RULE . . . . .	45
COMPUTED_VARIABLE_ROLES . . . . .	46
com_item_missingness . . . . .	46
com_qualified_item_missingness . . . . .	48
com_qualified_segment_missingness . . . . .	49
com_segment_missingness . . . . .	50
com_unit_missingness . . . . .	53
contradiction_functions_descriptions . . . . .	54
CONTRADICTION_TERM . . . . .	55
CONTRADICTION_TYPE . . . . .	55
con_contradictions . . . . .	56
con_contradictions_redcap . . . . .	58
con_inadmissible_categorical . . . . .	60
con_inadmissible_vocabulary . . . . .	61
con_limit_deviations . . . . .	63
dataquieR.acc_loess.exclude_constant_subgroups . . . . .	65
dataquieR.acc_loess.mark_time_points . . . . .	66
dataquieR.acc_loess.min_bw . . . . .	67
dataquieR.acc_loess.min_proportion . . . . .	68
dataquieR.acc_loess.plot_format . . . . .	68
dataquieR.acc_loess.plot_observations . . . . .	69
dataquieR.acc_margins_num . . . . .	70
dataquieR.acc_margins_sort . . . . .	71
dataquieR.acc_multivariate_outlier.scale . . . . .	72
dataquieR.applicability_problem . . . . .	72
dataquieR.col_con_con_empirical . . . . .	73

dataquieR.col_con_con_logical . . . . .	74
dataquieR.CONDITIONS_LEVEL_TRHESHOLD . . . . .	74
dataquieR.CONDITIONS_WITH_STACKTRACE . . . . .	75
dataquieR.convert_to_list_for_lapply . . . . .	76
dataquieR.debug . . . . .	77
dataquieR.des_summary_hard_lim_remove . . . . .	77
dataquieR.dontwrapresults . . . . .	78
dataquieR.droplevels_ReportSummaryTable . . . . .	79
dataquieR.dt_adjust . . . . .	80
dataquieR.ELEMENT_MISSMATCH_CHECKTYPE . . . . .	81
dataquieR.ERRORS_WITH_CALLER . . . . .	82
dataquieR.fix_column_type_on_read . . . . .	82
dataquieR.flip_mode . . . . .	83
dataquieR.force_item_specific_missing_codes . . . . .	84
dataquieR.force_label_col . . . . .	85
dataquieR.GAM_for_LOESS . . . . .	86
dataquieR.grading_formats . . . . .	86
dataquieR.grading_rulesets . . . . .	87
dataquieR.guess_character . . . . .	88
dataquieR.guess_missing_codes . . . . .	89
dataquieR.ignore_empty_vars . . . . .	90
dataquieR.intrinsic_applicability_problem . . . . .	90
dataquieR.lang . . . . .	91
dataquieR.lazy_plots . . . . .	92
dataquieR.lazy_plots_cache . . . . .	92
dataquieR.lazy_plots_gg_compatibility . . . . .	93
dataquieR.locale . . . . .	94
dataquieR.MAHALANOBIS_THRESHOLD . . . . .	95
dataquieR.max_cat_resp_var_levels_in_plot . . . . .	96
dataquieR.max_group_var_levels_in_plot . . . . .	96
dataquieR.max_group_var_levels_with_violins . . . . .	97
dataquieR.MAX_LABEL_LEN . . . . .	98
dataquieR.MAX_LONG_LABEL_LEN . . . . .	99
dataquieR.MAX_VALUE_LABEL_LEN . . . . .	99
dataquieR.MESSAGES_WITH_CALLER . . . . .	100
dataquieR.min_obs_per_group_var_in_plot . . . . .	101
dataquieR.min_time_points_for_cat_resp_var . . . . .	102
dataquieR.MULTIVARIATE_OUTLIER_CHECK . . . . .	103
dataquieR.non_disclosure . . . . .	104
dataquieR.old_factor_handling . . . . .	104
dataquieR.old_type_adjust . . . . .	105
dataquieR.precomputeStudyData . . . . .	106
dataquieR.print_block_load_factor . . . . .	107
dataquieR.progress_fkt_default . . . . .	108
dataquieR.progress_msg_fkt_default . . . . .	108
dataquieR.resume_checkpoint . . . . .	109
dataquieR.resume_print . . . . .	110
dataquieR.scale_level_heuristics_control_binaryrecodelimit . . . . .	111

dataquieR.scale_level_heuristics_control_metriclevels . . . . .	111
dataquieR.study_data_cache_max . . . . .	112
dataquieR.study_data_cache_metrics . . . . .	113
dataquieR.study_data_cache_metrics_env . . . . .	114
dataquieR.study_data_cache_metrics_env_default . . . . .	115
dataquieR.study_data_cache_quick_fill . . . . .	115
dataquieR.study_data_colnames_case_sensitive . . . . .	116
dataquieR.testdebug . . . . .	117
dataquieR.traceback . . . . .	118
dataquieR.type_adjust_parallel . . . . .	118
dataquieR.VALUE_LABELS_htmlescaped . . . . .	119
dataquieR.WARNINGS_WITH_CALLER . . . . .	120
dataquieR_resultset . . . . .	121
dataquieR_resultset2-class . . . . .	121
dataquieR_resultset_verify . . . . .	122
DATA_PREPARATION . . . . .	122
DATA_TYPES . . . . .	123
DATA_TYPES_OF_R_TYPE . . . . .	124
des_scatterplot_matrix . . . . .	124
des_summary . . . . .	125
des_summary_categorical . . . . .	127
des_summary_continuous . . . . .	128
DF_CODE . . . . .	130
DF_ELEMENT_COUNT . . . . .	130
DF_ID_REF_TABLE . . . . .	131
DF_ID_VARS . . . . .	131
DF_NAME . . . . .	132
DF_RECORD_CHECK . . . . .	132
DF_RECORD_COUNT . . . . .	133
DF_UNIQUE_ID . . . . .	133
DF_UNIQUE_ROWS . . . . .	134
dim.dataquieR_resultset2 . . . . .	134
dimensions . . . . .	135
dimnames.dataquieR_resultset2 . . . . .	135
dims . . . . .	136
DISTRIBUTIONS . . . . .	136
dq_report . . . . .	137
dq_report2 . . . . .	137
dq_report_by . . . . .	141
droplevels.ReportSummaryTable . . . . .	145
ggplotGrob.dq_lazy_ggplot . . . . .	145
GOLDSTANDARD . . . . .	146
grid.draw.util_pairs_ggplot_panels . . . . .	147
html_dependency_clipboard . . . . .	147
html_dependency_dataquieR . . . . .	148
html_dependency_jspdf . . . . .	148
html_dependency_report_dt . . . . .	149
html_dependency_tippy . . . . .	149

html_dependency_vert_dt . . . . .	149
int_all_datastructure_dataframe . . . . .	150
int_all_datastructure_segment . . . . .	151
int_datatype_matrix . . . . .	152
int_duplicate_content . . . . .	154
int_duplicate_ids . . . . .	155
int_encoding_errors . . . . .	156
int_part_vars_structure . . . . .	157
int_sts_element_dataframe . . . . .	158
int_sts_element_segment . . . . .	159
int_unexp_elements . . . . .	160
int_unexp_records_dataframe . . . . .	161
int_unexp_records_segment . . . . .	162
int_unexp_records_set . . . . .	164
IRV . . . . .	165
MAHALANOBIS_THRESHOLD . . . . .	165
MAXIMUM_LONG_STRING . . . . .	166
meta_data . . . . .	166
meta_data_computation . . . . .	167
meta_data_cross . . . . .	167
meta_data_dataframe . . . . .	167
meta_data_segment . . . . .	168
MISS_RESP . . . . .	168
MULTIVARIATE_OUTLIER_CHECK . . . . .	168
MULTIVARIATE_OUTLIER_CHECKTYPE . . . . .	169
nres . . . . .	170
pipeline_recursive_result . . . . .	170
pipeline_vectorized . . . . .	171
plot.dataquieR_summary . . . . .	171
prep_acc_distributions_with_ecdf . . . . .	172
prep_add_cause_label_df . . . . .	173
prep_add_computed_variables . . . . .	174
prep_add_data_frames . . . . .	175
prep_add_missing_codes . . . . .	176
prep_add_to_meta . . . . .	177
prep_apply_coding . . . . .	178
prep_check_for_dataquieR_updates . . . . .	179
prep_check_meta_data_dataframe . . . . .	180
prep_check_meta_data_segment . . . . .	181
prep_check_meta_names . . . . .	182
prep_clean_labels . . . . .	184
prep_combine_report_summaries . . . . .	185
prep_compare_meta_with_study . . . . .	186
prep_create_meta . . . . .	187
prep_create_meta_data_file . . . . .	188
prep_create_storr_factory . . . . .	189
prep_datatype_from_data . . . . .	189
prep_deparse_assignments . . . . .	190

prep_dq_data_type_of . . . . .	191
prep_expand_codes . . . . .	192
prep_extract_cause_label_df . . . . .	193
prep_extract_classes_by_functions . . . . .	194
prep_extract_summary . . . . .	194
prep_extract_summary.dataquieR_result . . . . .	195
prep_extract_summary.dataquieR_resultset2 . . . . .	196
prep_fix_meta_id_dups . . . . .	196
prep_get_data_frame . . . . .	198
prep_get_labels . . . . .	199
prep_get_study_data_segment . . . . .	201
prep_get_user_name . . . . .	202
prep_guess_encoding . . . . .	202
prep_link_escape . . . . .	203
prep_list_dataframes . . . . .	203
prep_list_voc . . . . .	204
prep_load_folder_with_metadata . . . . .	205
prep_load_report . . . . .	206
prep_load_report_from_backend . . . . .	206
prep_load_workbook_like_file . . . . .	207
prep_map_labels . . . . .	208
prep_merge_study_data . . . . .	209
prep_meta_data_v1_to_item_level_meta_data . . . . .	210
prep_min_obs_level . . . . .	211
prep_open_in_excel . . . . .	211
prep_pmap . . . . .	212
prep_prepare_dataframes . . . . .	213
prep_purge_data_frame_cache . . . . .	215
prep_realize_ggplot . . . . .	216
prep_remove_from_cache . . . . .	216
prep_render_pie_chart_from_summaryclasses_ggplot2 . . . . .	217
prep_render_pie_chart_from_summaryclasses_plotly . . . . .	218
prep_robust_guess_data_type . . . . .	219
prep_save_report . . . . .	219
prep_scalelevel_from_data_and_metadata . . . . .	220
prep_set_backend . . . . .	221
prep_study2meta . . . . .	221
prep_summary_to_classes . . . . .	222
prep_title_escape . . . . .	223
prep_undisclose . . . . .	223
prep_unsplit_val_tabs . . . . .	224
prep_valuelabels_from_data . . . . .	224
print.dataquieR_result . . . . .	225
print.dataquieR_resultset . . . . .	225
print.dataquieR_resultset2 . . . . .	226
print.dataquieR_summary . . . . .	227
print.DataSlot . . . . .	228
print.interval . . . . .	228

print.list . . . . .	229
print.master_result . . . . .	229
print.numeric_with_unit . . . . .	230
print.ReportSummaryTable . . . . .	230
print.Slot . . . . .	231
print.StudyDataSlot . . . . .	232
print.TableSlot . . . . .	232
print.util_pairs_ggplot_panels . . . . .	233
progress_init_fkt . . . . .	233
pro_applicability_matrix . . . . .	234
rbind.ReportSummaryTable . . . . .	236
RELCOMPL_SPEED . . . . .	236
REL_VAL . . . . .	237
resnames . . . . .	237
resnames.dataquieR_resultset2 . . . . .	238
RESPT_PER_ITEM . . . . .	238
SCALE_ACRONYM . . . . .	239
SCALE_LEVELS . . . . .	239
SCALE_NAME . . . . .	240
SEGMENT_ID_REF_TABLE . . . . .	241
SEGMENT_ID_TABLE . . . . .	241
SEGMENT_ID_VARS . . . . .	242
SEGMENT_MISS . . . . .	242
SEGMENT_PART_VARS . . . . .	243
SEGMENT_RECORD_CHECK . . . . .	243
SEGMENT_RECORD_COUNT . . . . .	244
SEGMENT_UNIQUE_ID . . . . .	244
SEGMENT_UNIQUE_ROWS . . . . .	245
SPLIT_CHAR . . . . .	245
study_data . . . . .	245
summary.dataquieR_resultset . . . . .	246
summary.dataquieR_resultset2 . . . . .	246
TOTRESPT . . . . .	247
UNITS . . . . .	247
UNIT_IS_COUNT . . . . .	248
UNIT_PREFIXES . . . . .	248
UNIT_PREFIX_FACTORS . . . . .	248
UNIT_SOURCES . . . . .	249
value/missing-lists . . . . .	249
VARATT_REQUIRE_LEVELS . . . . .	250
VARIABLE_LIST . . . . .	250
VARIABLE_LIST_ORDER . . . . .	251
VARIABLE_ROLES . . . . .	251
WELL_KNOWN_META_VARIABLE_NAMES . . . . .	252
[.dataquieR_resultset2 . . . . .	254
[<-.dataquieR_resultset2 . . . . .	255
[[.dataquieR_resultset2 . . . . .	256
[[<-.dataquieR_resultset2 . . . . .	256

\$ <code>.dataquieR_resultset2</code> . . . . .	257
\$ <code>&lt;-.dataquieR_resultset2</code> . . . . .	257
% <code>%numeric_with_unit</code> . . . . .	258
% <code>/%numeric_with_unit</code> . . . . .	258
<code>^.numeric_with_unit</code> . . . . .	259

**Index** **260**

\*.numeric\_with\_unit    *Operator caring for units*

**Description**

Operator caring for units

**Usage**

```
## S3 method for class 'numeric_with_unit'
e1 * e2
```

**Arguments**

e1	first argument
e2	second argument

**Value**

result

+`.numeric_with_unit`    *Operator caring for units*

**Description**

Operator caring for units

**Usage**

```
## S3 method for class 'numeric_with_unit'
e1 + e2
```

**Arguments**

e1	first argument
e2	second argument

**Value**

result

`-.numeric_with_unit`    *Operator caring for units*

---

**Description**

Operator caring for units

**Usage**

```
## S3 method for class 'numeric_with_unit'  
e1 - e2
```

**Arguments**

e1	first argument
e2	second argument

**Value**

result

---

`/.numeric_with_unit`    *Operator caring for units*

---

**Description**

Operator caring for units

**Usage**

```
## S3 method for class 'numeric_with_unit'  
e1 / e2
```

**Arguments**

e1	first argument
e2	second argument

**Value**

result

---

acc\_cat\_distributions *Plots and checks for distributions for categorical variables*

---

## Description

This function creates distribution plots for categorical variables.

[Descriptor](#)

## Usage

```
acc_cat_distributions(
  resp_vars = NULL,
  group_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2,
  n_cat_max = getOption("dataquieR.max_cat_resp_var_levels_in_plot",
    dataquieR.max_cat_resp_var_levels_in_plot_default),
  n_group_max = getOption("dataquieR.max_group_var_levels_in_plot",
    dataquieR.max_group_var_levels_in_plot_default),
  n_data_min = getOption("dataquieR.min_time_points_for_cat_resp_var",
    dataquieR.min_time_points_for_cat_resp_var_default)
)
```

## Arguments

resp_vars	<a href="#">variable</a> the name of the measurement variable
group_vars	<a href="#">variable</a> the name of the observer, device or reader variable
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
n_cat_max	maximum number of categories to be displayed individually for the categorical variable (resp_vars)
n_group_max	maximum number of categories to be displayed individually for the grouping variable (group_vars, devices / examiners)
n_data_min	minimum number of data points to create a time course plot for an individual category of the resp_vars variable

**Details**

To complete

**Value**

A [list](#) with:

- SummaryPlot: [ggplot2::ggplot](#) for the response variable in resp\_vars.

**See Also**

[Online Documentation](#)

---

acc\_distributions      *Plots and checks for distributions*

---

**Description**

Data quality indicator checks "Unexpected location" and "Unexpected proportion" with histograms.

[Indicator](#)

**Usage**

```
acc_distributions(  
  resp_vars = NULL,  
  study_data,  
  label_col,  
  item_level = "item_level",  
  check_param = c("any", "location", "proportion"),  
  plot_ranges = TRUE,  
  flip_mode = "noflip",  
  meta_data = item_level,  
  meta_data_v2  
)
```

**Arguments**

resp_vars	<a href="#">variable list</a> the names of the measurement variables
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
check_param	<a href="#">enum</a> any   location   proportion. Which type of check should be conducted (if possible): a check on the location of the mean or median value of the study data, a check on proportions of categories, or either of them if the necessary metadata is available.

plot_ranges	<b>logical</b> Should the plot show ranges and results from the data quality checks? (default: TRUE)
flip_mode	<b>enum</b> default   flip   noflip   auto. Should the plot be in default orientation, flipped, not flipped or auto-flipped. Not all options are always supported. In general, this can be controlled by setting the <code>roptions(dataquieR.flip_mode = ...)</code> . If called from <code>dq_report</code> , you can also pass <code>flip_mode</code> to all function calls or set them specifically using <code>specific_args</code> .
meta_data	<b>data.frame</b> old name for <code>item_level</code>
meta_data_v2	<b>character</b> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify <code>meta_data_v2</code> .

## Value

A [list](#) with:

- `SummaryTable`: [data.frame](#) containing data quality checks for "Unexpected location" (`FLG_acc_ud_loc`) and "Unexpected proportion" (`FLG_acc_ud_prop`) for each response variable in `resp_vars`.
- `SummaryData`: a [data.frame](#) containing data quality checks for "Unexpected location" and / or "Unexpected proportion" for a report
- `SummaryPlotList`: [list](#) of [ggplot2::ggplots](#) for each response variable in `resp_vars`.

## Algorithm of this implementation:

- If no response variable is defined, select all variables of type float or integer in the study data.
- Remove missing codes from the study data (if defined in the metadata).
- Remove measurements deviating from (hard) limits defined in the metadata (if defined).
- Exclude variables containing only NA or only one unique value (excluding NAs).
- Perform check for "Unexpected location" if defined in the metadata (needs a `LOCATION_METRIC` (mean or median) and `LOCATION_RANGE` (range of expected values for the mean and median, respectively)).
- Perform check for "Unexpected proportion" if defined in the metadata (needs `PROPORTION_RANGE` (range of expected values for the proportions of the categories)).
- Plot histogram(s).

## See Also

[Online Documentation](#)

---

 acc\_distributions\_ecdf

*ECDF plots for distribution checks*


---

## Description

Data quality indicator checks "Unexpected location" and "Unexpected proportion" if a grouping variable is included: Plots of empirical cumulative distributions for the subgroups.

### Descriptor

## Usage

```
acc_distributions_ecdf(
  resp_vars = NULL,
  group_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2,
  n_group_max = getOption("dataquieR.max_group_var_levels_in_plot",
    dataquieR.max_group_var_levels_in_plot_default),
  n_obs_per_group_min = getOption("dataquieR.min_obs_per_group_var_in_plot",
    dataquieR.min_obs_per_group_var_in_plot_default)
)
```

## Arguments

resp_vars	<a href="#">variable list</a> the names of the measurement variables
group_vars	<a href="#">variable list</a> the name of the observer, device or reader variable
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
n_group_max	maximum number of categories to be displayed individually for the grouping variable (group_vars, devices / examiners)
n_obs_per_group_min	minimum number of data points per group to create a graph for an individual category of the group_vars variable

**Value**

A [list](#) with:

- SummaryPlotList: [list](#) of [ggplot2::ggplots](#) for each response variable in resp\_vars.

**See Also**

[Online Documentation](#)

---

acc\_distributions\_loc *Plots and checks for distributions – Location*

---

**Description**

Data quality indicator checks "Unexpected location" and "Unexpected proportion" with histograms.

[Indicator](#)

**Usage**

```
acc_distributions_loc(
  resp_vars = NULL,
  study_data,
  label_col = VAR_NAMES,
  item_level = "item_level",
  check_param = "location",
  plot_ranges = TRUE,
  flip_mode = "noflip",
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

resp_vars	<a href="#">variable list</a> the names of the measurement variables
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
check_param	<a href="#">enum</a> any   location   proportion. Which type of check should be conducted (if possible): a check on the location of the mean or median value of the study data, a check on proportions of categories, or either of them if the necessary metadata is available.
plot_ranges	<a href="#">logical</a> Should the plot show ranges and results from the data quality checks? (default: TRUE)

flip_mode	<b>enum</b> default   flip   noflip   auto. Should the plot be in default orientation, flipped, not flipped or auto-flipped. Not all options are always supported. In general, this can be controlled by setting the <code>roptions(dataquieR.flip_mode = . . .)</code> . If called from <code>dq_report</code> , you can also pass <code>flip_mode</code> to all function calls or set them specifically using <code>specific_args</code> .
meta_data	<b>data.frame</b> old name for <code>item_level</code>
meta_data_v2	<b>character</b> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify <code>meta_data_v2</code> .

### Value

A [list](#) with:

- **SummaryTable**: [data.frame](#) containing data quality checks for "Unexpected location" (`FLG_acc_ud_loc`) and "Unexpected proportion" (`FLG_acc_ud_prop`) for each response variable in `resp_vars`.
- **SummaryData**: a [data.frame](#) containing data quality checks for "Unexpected location" and / or "Unexpected proportion" for a report
- **SummaryPlotList**: [list](#) of [ggplot2::ggplots](#) for each response variable in `resp_vars`.

### Algorithm of this implementation:

- If no response variable is defined, select all variables of type float or integer in the study data.
- Remove missing codes from the study data (if defined in the metadata).
- Remove measurements deviating from (hard) limits defined in the metadata (if defined).
- Exclude variables containing only NA or only one unique value (excluding NAs).
- Perform check for "Unexpected location" if defined in the metadata (needs a `LOCATION_METRIC` (mean or median) and `LOCATION_RANGE` (range of expected values for the mean and median, respectively)).
- Perform check for "Unexpected proportion" if defined in the metadata (needs `PROPORTION_RANGE` (range of expected values for the proportions of the categories)).
- Plot histogram(s).

### See Also

- [acc\\_distributions](#)
- [Online Documentation](#)

---

 acc\_distributions\_only

*Plots and checks for distributions – only*


---

## Description

### Descriptor

## Usage

```
acc_distributions_only(
  resp_vars = NULL,
  study_data,
  label_col = VAR_NAMES,
  item_level = "item_level",
  flip_mode = "noflip",
  meta_data = item_level,
  meta_data_v2
)
```

## Arguments

resp_vars	<a href="#">variable list</a> the names of the measurement variables
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
flip_mode	<a href="#">enum</a> default   flip   noflip   auto. Should the plot be in default orientation, flipped, not flipped or auto-flipped. Not all options are always supported. In general, this can be controlled by setting the <code>roptions(dataquieR.flip_mode = ...)</code> . If called from <code>dq_report</code> , you can also pass <code>flip_mode</code> to all function calls or set them specifically using <code>specific_args</code> .
meta_data	<a href="#">data.frame</a> old name for <code>item_level</code>
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify <code>meta_data_v2</code> .

## Value

A [list](#) with:

- `SummaryTable`: [data.frame](#) containing data quality checks for "Unexpected location" (FLG\_acc\_ud\_loc) and "Unexpected proportion" (FLG\_acc\_ud\_prop) for each response variable in `resp_vars`.
- `SummaryData`: a [data.frame](#) containing data quality checks for "Unexpected location" and / or "Unexpected proportion" for a report
- `SummaryPlotList`: [list](#) of [ggplot2::ggplots](#) for each response variable in `resp_vars`.

**Algorithm of this implementation:**

- If no response variable is defined, select all variables of type float or integer in the study data.
- Remove missing codes from the study data (if defined in the metadata).
- Remove measurements deviating from (hard) limits defined in the metadata (if defined).
- Exclude variables containing only NA or only one unique value (excluding NAs).
- Perform check for "Unexpected location" if defined in the metadata (needs a LOCATION\_METRIC (mean or median) and LOCATION\_RANGE (range of expected values for the mean and median, respectively)).
- Perform check for "Unexpected proportion" if defined in the metadata (needs PROPORTION\_RANGE (range of expected values for the proportions of the categories)).
- Plot histogram(s).

**See Also**

- [acc\\_distributions](#)
- [Online Documentation](#)

---

acc\_distributions\_prop

*Plots and checks for distributions – Proportion*

---

**Description**

Data quality indicator checks "Unexpected location" and "Unexpected proportion" with histograms.

[Indicator](#)

**Usage**

```
acc_distributions_prop(  
  resp_vars = NULL,  
  study_data,  
  label_col,  
  item_level = "item_level",  
  check_param = "proportion",  
  plot_ranges = TRUE,  
  flip_mode = "noflip",  
  meta_data = item_level,  
  meta_data_v2  
)
```

**Arguments**

resp_vars	<a href="#">variable list</a> the names of the measurement variables
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
check_param	<a href="#">enum</a> any   location   proportion. Which type of check should be conducted (if possible): a check on the location of the mean or median value of the study data, a check on proportions of categories, or either of them if the necessary metadata is available.
plot_ranges	<a href="#">logical</a> Should the plot show ranges and results from the data quality checks? (default: TRUE)
flip_mode	<a href="#">enum</a> default   flip   noflip   auto. Should the plot be in default orientation, flipped, not flipped or auto-flipped. Not all options are always supported. In general, this can be controlled by setting the <code>roptions(dataquieR.flip_mode = ...)</code> . If called from <code>dq_report</code> , you can also pass <code>flip_mode</code> to all function calls or set them specifically using <code>specific_args</code> .
meta_data	<a href="#">data.frame</a> old name for <code>item_level</code>
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify <code>meta_data_v2</code> .

**Value**

A [list](#) with:

- `SummaryTable`: [data.frame](#) containing data quality checks for "Unexpected location" (FLG\_acc\_ud\_loc) and "Unexpected proportion" (FLG\_acc\_ud\_prop) for each response variable in `resp_vars`.
- `SummaryData`: a [data.frame](#) containing data quality checks for "Unexpected location" and / or "Unexpected proportion" for a report
- `SummaryPlotList`: [list](#) of [ggplot2::ggplots](#) for each response variable in `resp_vars`.

**Algorithm of this implementation:**

- If no response variable is defined, select all variables of type float or integer in the study data.
- Remove missing codes from the study data (if defined in the metadata).
- Remove measurements deviating from (hard) limits defined in the metadata (if defined).
- Exclude variables containing only NA or only one unique value (excluding NAs).
- Perform check for "Unexpected location" if defined in the metadata (needs a `LOCATION_METRIC` (mean or median) and `LOCATION_RANGE` (range of expected values for the mean and median, respectively)).
- Perform check for "Unexpected proportion" if defined in the metadata (needs `PROPORTION_RANGE` (range of expected values for the proportions of the categories)).
- Plot histogram(s).

**See Also**

- [acc\\_distributions](#)
- [Online Documentation](#)

---

acc_end_digits	<i>Extension of <a href="#">acc_shape_or_scale</a> to examine uniform distributions of end digits</i>
----------------	---

---

**Description**

This implementation contrasts the empirical distribution of a measurement variables against assumed distributions. The approach is adapted from the idea of rootograms (Tukey (1977)) which is also applicable for count data (Kleiber and Zeileis (2016)).

**Indicator****Usage**

```
acc_end_digits(
  resp_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

resp_vars	<b>variable</b> the names of the measurement variables, mandatory
study_data	<b>data.frame</b> the data frame that contains the measurements
label_col	<b>variable attribute</b> the name of the column in the metadata with labels of variables
item_level	<b>data.frame</b> the data frame that contains metadata attributes of study data
meta_data	<b>data.frame</b> old name for item_level
meta_data_v2	<b>character</b> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.

**Value**

a **list** with:

- SummaryTable: **data.frame** with the columns Variables and FLG\_acc\_ud\_shape
- SummaryPlot: ggplot2 distribution plot comparing expected with observed distribution

**ALGORITHM OF THIS IMPLEMENTATION:**

- This implementation is restricted to data of type float or integer.
- Missing codes are removed from resp\_vars (if defined in the metadata)
- The user must specify the column of the metadata containing probability distribution (currently only: normal, uniform, gamma)
- Parameters of each distribution can be estimated from the data or are specified by the user
- A histogram-like plot contrasts the empirical vs. the technical distribution

**See Also**

[Online Documentation](#)

---

acc_loess	<i>Smooths and plots adjusted longitudinal measurements and longitudinal trends from logistic regression models</i>
-----------	---

---

**Description**

The following R implementation executes calculations for quality indicator "Unexpected location" (see [here](#)). Local regression (LOESS) is a versatile statistical method to explore an averaged course of time series measurements (Cleveland, Devlin, and Grosse 1988). In context of epidemiological data, repeated measurements using the same measurement device or by the same examiner can be considered a time series. LOESS allows to explore changes in these measurements over time.

[Descriptor](#)

**Usage**

```
acc_loess(
  resp_vars,
  group_vars = NULL,
  time_vars,
  co_vars = NULL,
  study_data,
  label_col = VAR_NAMES,
  item_level = "item_level",
  min_obs_in_subgroup = 30,
  resolution = 80,
  comparison_lines = list(type = c("mean/sd", "quartiles"), color = "grey30", linetype =
    2, sd_factor = 0.5),
  mark_time_points = getOption("dataquieR.acc_loess.mark_time_points",
    dataquieR.acc_loess.mark_time_points_default),
  plot_observations = getOption("dataquieR.acc_loess.plot_observations",
    dataquieR.acc_loess.plot_observations_default),
  plot_format = getOption("dataquieR.acc_loess.plot_format",
    dataquieR.acc_loess.plot_format_default),
```

```

meta_data = item_level,
meta_data_v2,
n_group_max = getOption("dataquieR.max_group_var_levels_in_plot",
  dataquieR.max_group_var_levels_in_plot_default),
enable_GAM = getOption("dataquieR.GAM_for_LOESS", dataquieR.GAM_for_LOESS_default),
exclude_constant_subgroups =
  getOption("dataquieR.acc_loess.exclude_constant_subgroups",
    dataquieR.acc_loess.exclude_constant_subgroups_default),
min_bandwidth = getOption("dataquieR.acc_loess.min_bw",
  dataquieR.acc_loess.min_bw_default),
min_proportion = getOption("dataquieR.acc_loess.min_proportion",
  dataquieR.acc_loess.min_proportion_default)
)

```

## Arguments

resp_vars	<b>variable</b> the name of the continuous measurement variable
group_vars	<b>variable</b> the name of the observer, device or reader variable
time_vars	<b>variable</b> the name of the variable giving the time of measurement
co_vars	<b>variable list</b> a vector of covariables for adjustment, for example age and sex. Can be NULL (default) for no adjustment.
study_data	<b>data.frame</b> the data frame that contains the measurements
label_col	<b>variable attribute</b> the name of the column in the metadata with labels of variables
item_level	<b>data.frame</b> the data frame that contains metadata attributes of study data
min_obs_in_subgroup	<b>integer</b> (optional argument) If group_vars is specified, this argument can be used to specify the minimum number of observations required for each of the subgroups. Subgroups with fewer observations are excluded. The default number is 30.
resolution	<b>numeric</b> the maximum number of time points used for plotting the trend lines
comparison_lines	<b>list</b> type and style of lines with which trend lines are to be compared. Can be mean +/- 0.5 standard deviation (the factor can be specified differently in sd_factor) or quartiles (Q1, Q2, and Q3). Arguments color and linetype are passed to <code>ggplot2::geom_line()</code> .
mark_time_points	<b>logical</b> mark time points with observations (caution, there may be many marks)
plot_observations	<b>logical</b> show observations as scatter plot in the background. If there are co_vars specified, the values of the observations in the plot will also be adjusted for the specified covariables.
plot_format	<b>enum</b> AUTO   COMBINED   FACETS   BOTH. Return the plot as one combined plot for all groups or as facet plots (one figure per group). BOTH will return both variants, AUTO will decide based on the number of observers.
meta_data	<b>data.frame</b> old name for item_level

meta_data_v2	<b>character</b> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
n_group_max	<b>integer</b> maximum number of categories to be displayed individually for the grouping variable (group_vars, devices / examiners)
enable_GAM	<b>logical</b> Can LOESS computations be replaced by general additive models to reduce memory consumption for large datasets?
exclude_constant_subgroups	<b>logical</b> Should subgroups with constant values be excluded?
min_bandwidth	<b>numeric</b> lower limit for the LOESS bandwidth, should be greater than 0 and less than or equal to 1. In general, increasing the bandwidth leads to a smoother trend line.
min_proportion	<b>numeric</b> lower limit for the proportion of the smaller group (cases or controls) for creating a LOESS figure, should be greater than 0 and less than 0.4.

## Details

If `mark_time_points` or `plot_observations` is selected, but would result in plotting more than 400 points, only a sample of the data will be displayed.

### Limitations

The application of LOESS requires model fitting, i.e. the smoothness of a model is subject to a smoothing parameter (span). Particularly in the presence of interval-based missing data, high variability of measurements combined with a low number of observations in one level of the `group_vars` may distort the fit. Since our approach handles data without knowledge of such underlying characteristics, finding the best fit is complicated if computational costs should be minimal. The default of LOESS in R uses a span of 0.75, which provides in most cases reasonable fits. The function `acc_loess` adapts the span for each level of the `group_vars` (with at least as many observations as specified in `min_obs_in_subgroup` and with at least three time points) based on the respective number of observations. LOESS consumes a lot of memory for larger datasets. That is why `acc_loess` switches to a generalized additive model with integrated smoothness estimation (gam by mgcv) if there are 1000 observations or more for at least one level of the `group_vars` (similar to `geom_smooth` from `ggplot2`).

## Value

a **list** with:

- `SummaryPlotList`: list with two plots if `plot_format = "BOTH"`, otherwise one of the two figures described below:
  - `Loess_fits_facets`: The plot contains LOESS-smoothed curves for each level of the `group_vars` in a separate panel. Added trend lines represent mean and standard deviation or quartiles (specified in `comparison_lines`) for moving windows over the whole data.
  - `Loess_fits_combined`: This plot combines all curves into one panel. Given a low number of levels in the `group_vars`, this plot eases comparisons. However, if the number increases this plot may be too crowded and unclear.

**See Also**

[Online Documentation](#)

---

acc_mahalanobis	<i>Calculate and plot Mahalanobis distances for social science indices</i>
-----------------	--

---

**Description**

A standard tool to calculate Mahalanobis distance. In this approach the Mahalanobis distance is calculated for ordinal variables (treated as continuous) to identify inattentive responses. It calculates the distance for each observational unit from the sample mean. The greater the distance, the atypical the responses.

[Indicator](#)

**Usage**

```
acc_mahalanobis(
  variable_group = NULL,
  label_col = VAR_NAMES,
  study_data,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2,
  mahalanobis_threshold =
    suppressWarnings(as.numeric(getOption("dataquieR.MAHALANOBIS_THRESHOLD",
    dataquieR.MAHALANOBIS_THRESHOLD_default)))
)
```

**Arguments**

`variable_group` [variable list](#) the names of the continuous measurement variables building a group, for that multivariate outliers make sense.

`label_col` [variable attribute](#) the name of the column in the metadata with labels of variables

`study_data` [data.frame](#) the data frame that contains the measurements

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`meta_data` [data.frame](#) old name for `item_level`

`meta_data_v2` [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

`mahalanobis_threshold`  
[numeric](#) TODO: ES

**Value**

a list with:

- SummaryTable: [data.frame](#) underlying the plot
- SummaryPlot: [ggplot2::ggplot2](#) outlier plot
- FlaggedStudyData [data.frame](#) contains the original data frame with the additional columns tukey, 3SD, hubert, and sigmagap. Every observation is coded 0 if no outlier was detected in the respective column and 1 if an outlier was detected. This can be used to exclude observations with outliers.

**ALGORITHM OF THIS IMPLEMENTATION:**

- Implementation is restricted to variables of type integer
- Remove missing codes from the study data (if defined in the metadata)
- The covariance matrix is estimated for all variables from `variable_group`
- The Mahalanobis distance of each observation is calculated  $MD_i^2 = (x_i - \mu)^T \Sigma^{-1} (x_i - \mu)$
- The default to consider a value an outlier is "use the 0.975 quantile of a chi-square distribution with p degrees of freedom" (Mayrhofer and Filzmoser, 2023) List function.

**See Also**

[Online Documentation](#)

---

acc\_margins

*Estimate marginal means, see [emmeans::emmeans](#)*

---

**Description**

This function examines the impact of so-called process variables on a measurement variable. This implementation combines a descriptive and a model-based approach. Process variables that can be considered in this implementation must be categorical. It is currently not possible to consider more than one process variable within one function call. The measurement variable can be adjusted for (multiple) covariables, such as age or sex, for example.

Marginal means rests on model-based results, i.e. a significantly different marginal mean depends on sample size. Particularly in large studies, small and irrelevant differences may become significant. The contrary holds if sample size is low.

[Indicator](#)

**Usage**

```
acc_margins(
  resp_vars = NULL,
  group_vars = NULL,
  co_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  threshold_type = "empirical",
  threshold_value,
  min_obs_in_subgroup = 5,
  min_obs_in_cat = 5,
  dichotomize_categorical_resp = TRUE,
  cut_off_linear_model_for_ord = 10,
  meta_data = item_level,
  meta_data_v2,
  sort_group_var_levels = getOption("dataquieR.acc_margins_sort",
    dataquieR.acc_margins_sort_default),
  include_numbers_in_figures = getOption("dataquieR.acc_margins_num",
    dataquieR.acc_margins_num_default),
  n_violin_max = getOption("dataquieR.max_group_var_levels_with_violins",
    dataquieR.max_group_var_levels_with_violins_default)
)
```

**Arguments**

resp_vars	<b>variable</b> the name of the measurement variable
group_vars	<b>variable list</b> len=1-1. the name of the observer, device or reader variable
co_vars	<b>variable list</b> a vector of covariables, e.g. age and sex for adjustment
study_data	<b>data.frame</b> the data frame that contains the measurements
label_col	<b>variable attribute</b> the name of the column in the metadata with labels of variables
item_level	<b>data.frame</b> the data frame that contains metadata attributes of study data
threshold_type	<b>enum</b> empirical   user   none. In case empirical is chosen, a multiplier of the scale measure is used. In case of user, a value of the mean or probability (binary data) has to be defined see <b>Implementation and use of thresholds</b> in the online documentation). In case of none, no thresholds are displayed and no flagging of unusual group levels is applied.
threshold_value	<b>numeric</b> a multiplier or absolute value (see <b>Implementation and use of thresholds</b> in the online documentation).
min_obs_in_subgroup	<b>integer</b> from=0. This optional argument specifies the minimum number of observations that is required to include a subgroup (level) of the group_var in the analysis. Subgroups with less observations are excluded.
min_obs_in_cat	<b>integer</b> This optional argument specifies the minimum number of observations that is required to include a category (level) of the outcome (resp_vars) in the

analysis. Categories with less observations are combined into one group. If the collapsed category contains less observations than required, it will be excluded from the analysis.

dichotomize\_categorical\_resp

**logical** Should nominal response variables always be transformed to binary variables?

cut\_off\_linear\_model\_for\_ord

**integer** from=0. This optional argument specifies the minimum number of observations for individual levels of an ordinal outcome (resp\_var) that is required to run a linear model instead of an ordered regression (i.e., a cut-off value above which linear models are considered a good approximation). The argument can be set to NULL if ordered regression models are preferred for ordinal data in any case.

meta\_data **data.frame** old name for item\_level

meta\_data\_v2 **character** path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify meta\_data\_v2.

sort\_group\_var\_levels

**logical** Should the levels of the grouping variable be sorted descending by the number of observations? Note that ordinal grouping variables will not be re-ordered.

include\_numbers\_in\_figures

**logical** Should the figure report the number of observations for each level of the grouping variable?

n\_violin\_max **integer** from=0. This optional argument specifies the maximum number of levels of the group\_var for which violin plots will be shown in the figure.

## Details

### Limitations

Selecting the appropriate distribution is complex. Dozens of continuous, discrete or mixed distributions are conceivable in the context of epidemiological data. Their exact exploration is beyond the scope of this data quality approach. The present function uses the help function `util_dist_selection`, the assigned `SCALE_LEVEL` and the `DATA_TYPE` to discriminate the following cases:

- continuous data
- binary data
- count data with  $\leq 20$  distinct values
- count data with  $> 20$  distinct values (treated as continuous)
- nominal data
- ordinal data

Continuous data and count data with more than 20 distinct values are analyzed by linear models. Count data with up to 20 distinct values are modeled by a Poisson regression. For binary data, the implementation uses logistic regression. Nominal response variables will either be transformed to binary variables or analyzed by multinomial logistic regression models. The latter option is

only available if the argument `dichotomize_categorical_resp` is set to `FALSE` and if the package `nnet` is installed. The transformation to a binary variable can be user-specified using the metadata columns `RECODE_CASES` and/or `RECODE_CONTROL`. Otherwise, the most frequent category will be assigned to cases and the remaining categories to control. For ordinal response variables, the argument `cut_off_linear_model_for_ord` controls whether the data is analyzed in the same way as continuous data: If every level of the variable has at least as many observations as specified in the argument, the data will be analyzed by a linear model. Otherwise, the data will be modeled by a ordered regression, if the package `ordinal` is installed.

## Value

a list with:

- `SummaryTable`: `data.frame` underlying the plot
- `ResultData`: `data.frame`
- `SummaryPlot`: `ggplot2::ggplot()` margins plot

## See Also

[Online Documentation](#)

---

acc\_multivariate\_outlier

*Calculate and plot Mahalanobis distances*

---

## Description

A standard tool to detect multivariate outliers is the Mahalanobis distance. This approach is very helpful for the interpretation of the plausibility of a measurement given the value of another. In this approach the Mahalanobis distance is used as a univariate measure itself. We apply the same rules for the identification of outliers as in univariate outliers:

- the classical approach from Tukey:  $1.5 * IQR$  from the 1st ( $Q_{25}$ ) or 3rd ( $Q_{75}$ ) quartile.
- the 3SD approach, i.e. any measurement of the Mahalanobis distance not in the interval of  $\bar{x} \pm 3 * \sigma$  is considered an outlier.
- the approach from Hubert for skewed distributions which is embedded in the R package **robustbase**
- a completely heuristic approach named  $\sigma$ -gap.

For further details, please see the vignette for univariate outlier.

[Indicator](#)

**Usage**

```
acc_multivariate_outlier(
  variable_group = NULL,
  id_vars = NULL,
  label_col = VAR_NAMES,
  study_data,
  item_level = "item_level",
  n_rules = 4,
  max_non_outliers_plot = 10000,
  criteria = c("tukey", "3sd", "hubert", "sigmagap"),
  meta_data = item_level,
  meta_data_v2,
  scale = getOption("dataquieR.acc_multivariate_outlier.scale",
    dataquieR.acc_multivariate_outlier.scale_default),
  multivariate_outlier_check = TRUE
)
```

**Arguments**

`variable_group` [variable list](#) the names of the continuous measurement variables building a group, for that multivariate outliers make sense.

`id_vars` [variable](#) optional, an ID variable of the study data. If not specified row numbers are used.

`label_col` [variable attribute](#) the name of the column in the metadata with labels of variables

`study_data` [data.frame](#) the data frame that contains the measurements

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`n_rules` [numeric](#) from=1 to=4. the no. of rules that must be violated to classify as outlier

`max_non_outliers_plot` [integer](#) from=0. Maximum number of non-outlier points to be plot. If more points exist, a subsample will be plotted only. Note, that sampling is not deterministic.

`criteria` [set](#) tukey | 3SD | hubert | sigmagap. a vector with methods to be used for detecting outliers.

`meta_data` [data.frame](#) old name for item\_level

`meta_data_v2` [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify meta\_data\_v2.

`scale` [logical](#) Should min-max-scaling be applied per variable?

`multivariate_outlier_check` [logical](#) really check, pipeline use, only.

**Value**

a list with:

- SummaryTable: [data.frame](#) underlying the plot

- SummaryPlot: [ggplot2::ggplot2](#) outlier plot
- FlaggedStudyData [data.frame](#) contains the original data frame with the additional columns `tukey`, `3SD`, `hubert`, and `sigmagap`. Every observation is coded 0 if no outlier was detected in the respective column and 1 if an outlier was detected. This can be used to exclude observations with outliers.

#### ALGORITHM OF THIS IMPLEMENTATION:

- Implementation is restricted to variables of type float
- Remove missing codes from the study data (if defined in the metadata)
- The covariance matrix is estimated for all variables from `variable_group`
- The Mahalanobis distance of each observation is calculated  $MD_i^2 = (x_i - \mu)^T \Sigma^{-1} (x_i - \mu)$
- The four rules mentioned above are applied on this distance for each observation in the study data
- An output data frame is generated that flags each outlier
- A parallel coordinate plot indicates respective outliers

List function.

#### See Also

[Online Documentation](#)

---

acc\_robust\_univariate\_outlier

*Identify univariate outliers by four different approaches*

---

#### Description

A classical but still popular approach to detect univariate outlier is the boxplot method introduced by Tukey 1977. The boxplot is a simple graphical tool to display information about continuous univariate data (e.g., median, lower and upper quartile). Outliers are defined as values deviating more than  $1.5 \times IQR$  from the 1st (Q25) or 3rd (Q75) quartile. The strength of Tukey's method is that it makes no distributional assumptions and thus is also applicable to skewed or non mound-shaped data Marsh and Seo, 2006. Nevertheless, this method tends to identify frequent measurements which are falsely interpreted as true outliers.

A somewhat more conservative approach in terms of symmetric and/or normal distributions is the 3SD approach, i.e. any measurement not in the interval of  $mean(x) + / - 3 * \sigma$  is considered an outlier.

Both methods mentioned above are not ideally suited to skewed distributions. As many biomarkers such as laboratory measurements represent in skewed distributions the methods above may be insufficient. The approach of Hubert and Vandervieren 2008 adjusts the boxplot for the skewness of the distribution. This approach is implemented in several R packages such as [robustbase::mc](#) which is used in this implementation of [dataquieR](#).

Another completely heuristic approach is also included to identify outliers. The approach is based on the assumption that the distances between measurements of the same underlying distribution should be homogeneous. For comprehension of this approach:

- consider an ordered sequence of all measurements.
- between these measurements all distances are calculated.
- the occurrence of larger distances between two neighboring measurements may than indicate a distortion of the data. For the heuristic definition of a large distance  $1 * \sigma$  has been been chosen.

Note, that the plots are not deterministic, because they use [ggplot2::geom\\_jitter](#).

#### Indicator

#### Usage

```
acc_robust_univariate_outlier(
  resp_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  exclude_roles,
  n_rules = length(unique(criteria)),
  max_non_outliers_plot = 10000,
  criteria = c("tukey", "3sd", "hubert", "sigmagap"),
  meta_data = item_level,
  meta_data_v2
)
```

#### Arguments

resp_vars	<a href="#">variable list</a> the name of the continuous measurement variable
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
exclude_roles	<a href="#">variable roles</a> a character (vector) of variable roles not included
n_rules	<a href="#">integer</a> from=1 to=4. the no. rules that must be violated to flag a variable as containing outliers. The default is 4, i.e. all.
max_non_outliers_plot	<a href="#">integer</a> from=0. Maximum number of non-outlier points to be plot. If more points exist, a subsample will be plotted only. Note, that sampling is not deterministic.
criteria	<a href="#">set</a> tukey   3SD   hubert   sigmagap. a vector with methods to be used for detecting outliers.
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.

#### Details

**Hint:** *The function is designed for unimodal data only.*

**Value**

a list with:

- SummaryTable: `data.frame` with the columns Variables, Mean, SD, Median, Skewness, Tukey (N), 3SD (N), Hubert (N), Sigma-gap (N), NUM\_acc\_ud\_outlu, Outliers, low (N), Outliers, high (N) Grading
  - SummaryData: `data.frame` with the columns Variables, Mean, SD, Median, Skewness, Tukey (N), 3SD (N), Hubert (N), Sigma-gap (N), Outliers (N), Outliers, low (N), Outliers, high (N)
  - SummaryPlotList: `ggplot2::ggplot` univariate outlier plots

**ALGORITHM OF THIS IMPLEMENTATION:**

- Select all variables of type float in the study data
- Remove missing codes from the study data (if defined in the metadata)
- Remove measurements deviating from limits defined in the metadata
- Identify outliers according to the approaches of Tukey (Tukey 1977), 3SD (Saleem et al. 2021), Hubert (Hubert and Vandervieren 2008), and SigmaGap (heuristic)
- An output data frame is generated which indicates the no. possible outliers, the direction of deviations (Outliers, low; Outliers, high) for all methods and a summary score which sums up the deviations of the different rules
- A scatter plot is generated for all examined variables, flagging observations according to the no. violated rules (step 5).

**See Also**

[acc\\_univariate\\_outlier](#)

---

acc\_shape\_or\_scale      *Compare observed versus expected distributions*

---

**Description**

This implementation contrasts the empirical distribution of a measurement variables against assumed distributions. The approach is adapted from the idea of rootograms (Tukey 1977) which is also applicable for count data (Kleiber and Zeileis 2016).

[Indicator](#)

**Usage**

```
acc_shape_or_scale(
  resp_vars,
  study_data,
  label_col,
  item_level = "item_level",
  dist_col,
  guess,
  par1,
  par2,
  end_digits,
  flip_mode = "noflip",
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

resp_vars	<a href="#">variable</a> the name of the continuous measurement variable
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
dist_col	<a href="#">variable attribute</a> the name of the variable attribute in meta_data that provides the expected distribution of a study variable
guess	<a href="#">logical</a> estimate parameters
par1	<a href="#">numeric</a> first parameter of the distribution if applicable
par2	<a href="#">numeric</a> second parameter of the distribution if applicable
end_digits	<a href="#">logical</a> internal use. check for end digits preferences
flip_mode	<a href="#">enum</a> default   flip   noflip   auto. Should the plot be in default orientation, flipped, not flipped or auto-flipped. Not all options are always supported. In general, this can be controlled by setting the <code>roptions(dataquieR.flip_mode = ...)</code> . If called from <code>dq_report</code> , you can also pass <code>flip_mode</code> to all function calls or set them specifically using <code>specific_args</code> .
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify <code>meta_data_v2</code> .

**Value**

a list with:

- ResultData: [data.frame](#) underlying the plot
- SummaryPlot: [ggplot2::ggplot2](#) probability distribution plot
- SummaryTable: [data.frame](#) with the columns Variables and FLG\_acc\_ud\_shape

**ALGORITHM OF THIS IMPLEMENTATION:**

- This implementation is restricted to data of type float or integer.
- Missing codes are removed from resp\_vars (if defined in the metadata)
- The user must specify the column of the metadata containing probability distribution (currently only: normal, uniform, gamma)
- Parameters of each distribution can be estimated from the data or are specified by the user
- A histogram-like plot contrasts the empirical vs. the technical distribution

**See Also**

[Online Documentation](#)

---

acc\_univariate\_outlier

*Identify univariate outliers by four different approaches*

---

**Description**

A classical but still popular approach to detect univariate outlier is the boxplot method introduced by Tukey 1977. The boxplot is a simple graphical tool to display information about continuous univariate data (e.g., median, lower and upper quartile). Outliers are defined as values deviating more than  $1.5 \times IQR$  from the 1st (Q25) or 3rd (Q75) quartile. The strength of Tukey's method is that it makes no distributional assumptions and thus is also applicable to skewed or non mound-shaped data Marsh and Seo, 2006. Nevertheless, this method tends to identify frequent measurements which are falsely interpreted as true outliers.

A somewhat more conservative approach in terms of symmetric and/or normal distributions is the 3SD approach, i.e. any measurement not in the interval of  $mean(x) + / - 3 * \sigma$  is considered an outlier.

Both methods mentioned above are not ideally suited to skewed distributions. As many biomarkers such as laboratory measurements represent in skewed distributions the methods above may be insufficient. The approach of Hubert and Vandervieren 2008 adjusts the boxplot for the skewness of the distribution. This approach is implemented in several R packages such as `robustbase::mc` which is used in this implementation of `dataquieR`.

Another completely heuristic approach is also included to identify outliers. The approach is based on the assumption that the distances between measurements of the same underlying distribution should be homogeneous. For comprehension of this approach:

- consider an ordered sequence of all measurements.
- between these measurements all distances are calculated.
- the occurrence of larger distances between two neighboring measurements may indicate a distortion of the data. For the heuristic definition of a large distance  $1 * \sigma$  has been chosen.

Note, that the plots are not deterministic, because they use `ggplot2::geom_jitter`.

[Indicator](#)

**Usage**

```
acc_univariate_outlier(
  resp_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  exclude_roles,
  n_rules = length(unique(criteria)),
  max_non_outliers_plot = 10000,
  criteria = c("tukey", "3sd", "hubert", "sigmagap"),
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

`resp_vars` [variable list](#) the name of the continuous measurement variable

`study_data` [data.frame](#) the data frame that contains the measurements

`label_col` [variable attribute](#) the name of the column in the metadata with labels of variables

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`exclude_roles` [variable roles](#) a character (vector) of variable roles not included

`n_rules` [integer](#) from=1 to=4. the no. rules that must be violated to flag a variable as containing outliers. The default is 4, i.e. all.

`max_non_outliers_plot` [integer](#) from=0. Maximum number of non-outlier points to be plot. If more points exist, a subsample will be plotted only. Note, that sampling is not deterministic.

`criteria` [set](#) tukey | 3SD | hubert | sigmagap. a vector with methods to be used for detecting outliers.

`meta_data` [data.frame](#) old name for item\_level

`meta_data_v2` [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify meta\_data\_v2.

**Details**

**Hint:** *The function is designed for unimodal data only.*

**Value**

a list with:

- SummaryTable: [data.frame](#) with the columns Variables, Mean, SD, Median, Skewness, Tukey (N), 3SD (N), Hubert (N), Sigma-gap (N), NUM\_acc\_ud\_outlu, Outliers, low (N), Outliers, high (N) Grading

- SummaryData: `data.frame` with the columns Variables, Mean, SD, Median, Skewness, Tukey (N), 3SD (N), Hubert (N), Sigma-gap (N), Outliers (N), Outliers, low (N), Outliers, high (N)
- SummaryPlotList: `ggplot2::ggplot` univariate outlier plots

#### ALGORITHM OF THIS IMPLEMENTATION:

- Select all variables of type float in the study data
- Remove missing codes from the study data (if defined in the metadata)
- Remove measurements deviating from limits defined in the metadata
- Identify outliers according to the approaches of Tukey (Tukey 1977), 3SD (Saleem et al. 2021), Hubert (Hubert and Vandervieren 2008), and SigmaGap (heuristic)
- An output data frame is generated which indicates the no. possible outliers, the direction of deviations (Outliers, low; Outliers, high) for all methods and a summary score which sums up the deviations of the different rules
- A scatter plot is generated for all examined variables, flagging observations according to the no. violated rules (step 5).

#### See Also

- [acc\\_robust\\_univariate\\_outlier](#)
- [Online Documentation](#)

---

acc\_varcomp

*Utility function to compute model-based ICC depending on the (statistical) data type*

---

#### Description

This function is still under construction. It is designed to run for any statistical data type as follows:

- Variables with only two distinct values will be modeled by mixed effects logistic regression.
- Nominal variables will be transformed to binary variables. This can be user-specified using the metadata columns RECODE\_CASES and/or RECODE\_CONTROL. Otherwise, the most frequent category will be assigned to cases and the remaining categories to control. As for other binary variables, the ICC will be computed using a mixed effects logistic regression.
- Ordinal variables will be analyzed by linear mixed effects models, if every level of the variable has at least as many observations as specified in the argument `cut_off_linear_model_for_ord`. Otherwise, the data will be modeled by a mixed effects ordered regression, if the package `ordinal` is available.
- Metric variables with integer values are analyzed by linear mixed effects models.
- For variables with data type `float`, the existing implementation `acc_varcomp` is called, which also uses linear mixed effects models.

[Indicator](#)

**Usage**

```
acc_varcomp(
  resp_vars = NULL,
  group_vars = NULL,
  co_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  min_obs_in_subgroup = 10,
  min_subgroups = 5,
  cut_off_linear_model_for_ord = 10,
  threshold_value = lifecycle::deprecated(),
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

`resp_vars` [variable](#) the name of the measurement variable

`group_vars` [variable](#) the name of the examiner, device or reader variable

`co_vars` [variable list](#) a vector of covariables, e.g. age and sex, for adjustment

`study_data` [data.frame](#) the data frame that contains the measurements

`label_col` [variable attribute](#) the name of the column in the metadata with labels of variables

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`min_obs_in_subgroup` [integer](#) from=0. This optional argument specifies the minimum number of observations that is required to include a subgroup (level) of the `group_var` in the analysis. Subgroups with less observations are excluded.

`min_subgroups` [integer](#) from=0. This optional argument specifies the minimum number of subgroups (level) of the `group_var` that is required to run the analysis. If there are less subgroups, the analysis is not conducted.

`cut_off_linear_model_for_ord` [integer](#) from=0. This optional argument specifies the minimum number of observations for individual levels of an ordinal outcome (`resp_var`) that is required to run a linear mixed effects model instead of a mixed effects ordered regression (i.e., a cut-off value above which linear models are considered a good approximation). The argument can be set to `NULL` if ordered regression models are preferred for ordinal data in any case.

`threshold_value` [Deprecated](#).

`meta_data` [data.frame](#) old name for `item_level`

`meta_data_v2` [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

**Details**

Not yet described

**Value**

The function returns two data frames, 'SummaryTable' and 'SummaryData', that differ only in the names of the columns.

---

as.character.interval as.character *implementation for the class interval*

---

**Description**

such objects, for now, only occur in RECCap rules, so this function is meant for internal use, mostly – for now.

**Usage**

```
## S3 method for class 'interval'  
as.character(x, ...)
```

**Arguments**

x	interval objects to convert
...	not used yet

**Value**

interval as character

**See Also**

base::as.character

---

```
as.data.frame.dataquieR_resultset
```

*Convert a full dataquieR report to a data.frame*

---

### Description

Deprecated

### Usage

```
## S3 method for class 'dataquieR_resultset'  
as.data.frame(x, ...)
```

### Arguments

x	Deprecated
...	Deprecated

### Value

Deprecated

---

```
as.list.dataquieR_resultset
```

*Convert a full dataquieR report to a list*

---

### Description

Deprecated

### Usage

```
## S3 method for class 'dataquieR_resultset'  
as.list(x, ...)
```

### Arguments

x	Deprecated
...	Deprecated

### Value

Deprecated

---

```
as.list.dataquieR_resultset2
```

*inefficient way to convert a report to a list. try [prep\\_set\\_backend\(\)](#)*

---

**Description**

inefficient way to convert a report to a list. try [prep\\_set\\_backend\(\)](#)

**Usage**

```
## S3 method for class 'dataquieR_resultset2'
as.list(x, ...)
```

**Arguments**

x	<a href="#">dataquieR_resultset2</a>
...	no used

**Value**

[list](#)

---

ASSOCIATION\_DIRECTION *Cross-item level metadata attribute name*

---

**Description**

The allowable direction of an association. The input is a string that can be either "positive" or "negative".

**Usage**

```
ASSOCIATION_DIRECTION
```

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_cross](#)

Other [meta\\_data\\_cross](#): [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

---

ASSOCIATION_FORM	<i>Cross-item level metadata attribute name</i>
------------------	---

---

**Description**

The allowable form of association. The string specifies the form based on a selected list.

**Usage**

ASSOCIATION\_FORM

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_cross](#)

Other [meta\\_data\\_cross](#): [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

---

ASSOCIATION_METRIC	<i>Cross-item level metadata attribute name</i>
--------------------	---

---

**Description**

The metric underlying the association in [ASSOCIATION\\_RANGE](#). The input is a string that specifies the analysis algorithm to be used.

**Usage**

ASSOCIATION\_METRIC

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_cross](#)

Other [meta\\_data\\_cross](#): [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

---

ASSOCIATION_RANGE	<i>Cross-item level metadata attribute name</i>
-------------------	---

---

**Description**

Specifies the allowable range of an association. The inclusion of the endpoints follows standard mathematical notation using round brackets for open intervals and square brackets for closed intervals. Values must be separated by a semicolon.

**Usage**

ASSOCIATION\_RANGE

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_cross](#)

Other [meta\\_data\\_cross](#): [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

---

CHECK_ID	<i>Cross-item level metadata attribute name</i>
----------	---

---

**Description**

Specifies the unique IDs for cross-item level metadata records

**Usage**

CHECK\_ID

**Format**

An object of class character of length 1.

**Details**

if missing, dataquieR will create such IDs

**See Also**

[meta\\_data\\_cross](#)

Other `meta_data_cross`: ASSOCIATION\_DIRECTION, ASSOCIATION\_FORM, ASSOCIATION\_METRIC, ASSOCIATION\_RANGE, CHECK\_LABEL, COMPUTED\_VARIABLE\_ROLES, CONTRADICTION\_TERM, CONTRADICTION\_TYPE, DATA\_PREPARATION, GOLDSTANDARD, IRV, MAHALANOBIS\_THRESHOLD, MAXIMUM\_LONG\_STRING, MISS\_RESP, MULTIVARIATE\_OUTLIER\_CHECK, MULTIVARIATE\_OUTLIER\_CHECKTYPE, RELCOMPL\_SPEED, REL\_VAL, RESPT\_PER\_ITEM, SCALE\_ACRONYM, SCALE\_NAME, TOTRESPT, VARIABLE\_LIST, VARIABLE\_LIST\_ORDER, `meta_data_computation`, `meta_data_cross`

---

CHECK\_LABEL

*Cross-item level metadata attribute name*

---

**Description**

Specifies the unique labels for cross-item level metadata records

**Usage**

CHECK\_LABEL

**Format**

An object of class character of length 1.

**Details**

if missing, dataquieR will create such labels

**See Also**

[meta\\_data\\_cross](#)

Other `meta_data_cross`: ASSOCIATION\_DIRECTION, ASSOCIATION\_FORM, ASSOCIATION\_METRIC, ASSOCIATION\_RANGE, CHECK\_ID, COMPUTED\_VARIABLE\_ROLES, CONTRADICTION\_TERM, CONTRADICTION\_TYPE, DATA\_PREPARATION, GOLDSTANDARD, IRV, MAHALANOBIS\_THRESHOLD, MAXIMUM\_LONG\_STRING, MISS\_RESP, MULTIVARIATE\_OUTLIER\_CHECK, MULTIVARIATE\_OUTLIER\_CHECKTYPE, RELCOMPL\_SPEED, REL\_VAL, RESPT\_PER\_ITEM, SCALE\_ACRONYM, SCALE\_NAME, TOTRESPT, VARIABLE\_LIST, VARIABLE\_LIST\_ORDER, `meta_data_computation`, `meta_data_cross`

---

check_table	<i>Data frame with contradiction rules</i>
-------------	--

---

**Description**

Two versions exist, the newer one is used by [con\\_contradictions\\_redcap](#) and is described [here](#)., the older one used by [con\\_contradictions](#) is described [here](#).

**See Also**

[meta\\_data\\_cross](#)

---

CODE_CLASSES	<i>types of value codes</i>
--------------	-----------------------------

---

**Description**

types of value codes

**Usage**

CODE\_CLASSES

**Format**

An object of class list of length 3.

---

CODE_LIST_TABLE	<i>Default Name of the Table featuring Code Lists</i>
-----------------	---

---

**Description**

Default Name of the Table featuring Code Lists

Metadata sheet name containing VALUE\_LABEL\_TABLES This metadata sheet can contain both value labels of several VALUE\_LABEL\_TABLE and also Missing and JUMP tables

**Usage**

CODE\_LIST\_TABLE

CODE\_LIST\_TABLE

**Format**

An object of class character of length 1.

An object of class character of length 1.

---

CODE_ORDER	<i>Only existence is checked, order not yet used</i>
------------	--

---

**Description**

Only existence is checked, order not yet used

**Usage**

CODE\_ORDER

**Format**

An object of class character of length 1.

---

COMPUTATION_RULE	<i>Cross-item level metadata attribute name</i>
------------------	---

---

**Description**

Cross-item level metadata attribute name

**Usage**

COMPUTATION\_RULE

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_computation](#)

---

**COMPUTED\_VARIABLE\_ROLES**

*SSI related Cross-item level metadata attribute names Computed Variable roles can be one of the following:*

---

**Description**

- MAXIMUM\_LONG\_STRING Social Science: Computed Indicator Variable, maximum long string
- IRV Social Science: Computed Indicator Variable, IRV
- TOTRESPT Social Science: Computed Indicator Variable, TOTRESPT
- RESPT\_PER\_ITEM Social Science: Computed Indicator Variable, RESPT\_PER\_ITEM
- RELCOMPL\_SPEED Social Science: Computed Indicator Variable, RELCOMPL\_SPEED
- MISS\_RESP Social Science: Computed Indicator Variable, MISS\_RESP
- NA Social Science: Computed Indicator Variable – N/A

**See Also**
[VARIABLE\\_ROLES](#)

Other meta\_data\_cross: [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

Other SSI: [IRV](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [RELCOMPL\\_SPEED](#), [RESPT\\_PER\\_ITEM](#), [TOTRESPT](#)

---

com\_item\_missingness *Summarize missingness columnwise (in variable)*

---

**Description**

Item-Missingness (also referred to as item nonresponse (De Leeuw et al. 2003)) describes the missingness of single values, e.g. blanks or empty data cells in a data set. Item-Missingness occurs for example in case a respondent does not provide information for a certain question, a question is overlooked by accident, a programming failure occurs or a provided answer were missed while entering the data.

[Indicator](#)

**Usage**

```

com_item_missingness(
  resp_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  show_causes = TRUE,
  cause_label_df,
  include_sysmiss = TRUE,
  threshold_value,
  suppressWarnings = FALSE,
  assume_consistent_codes = TRUE,
  expand_codes = assume_consistent_codes,
  drop_levels = FALSE,
  expected_observations = c("HIERARCHY", "ALL", "SEGMENT"),
  pretty_print = lifecycle::deprecated(),
  meta_data = item_level,
  meta_data_v2
)

```

**Arguments**

**resp\_vars** [variable list](#) the name of the measurement variables

**study\_data** [data.frame](#) the data frame that contains the measurements

**label\_col** [variable attribute](#) the name of the column in the metadata with labels of variables

**item\_level** [data.frame](#) the data frame that contains metadata attributes of study data

**show\_causes** [logical](#) if TRUE, then the distribution of missing codes is shown

**cause\_label\_df** [data.frame](#) missing code table. If missing codes have labels the respective data frame can be specified here or in the metadata as assignments, see [cause\\_label\\_df](#)

**include\_sysmiss** [logical](#) Optional, if TRUE system missingness (NAs) is evaluated in the summary plot

**threshold\_value** [numeric](#) from=0 to=100. a numerical value ranging from 0-100

**suppressWarnings** [logical](#) warn about consistency issues with missing and jump lists

**assume\_consistent\_codes** [logical](#) if TRUE and no labels are given and the same missing/jump code is used for more than one variable, the labels assigned for this code are treated as being the same for all variables.

**expand\_codes** [logical](#) if TRUE, code labels are copied from other variables, if the code is the same and the label is set somewhere

**drop\_levels** [logical](#) if TRUE, do not display unused missing codes in the figure legend.

**expected\_observations** [enum](#) HIERARCHY | ALL | SEGMENT. If ALL, all observations are expected to comprise all study segments. If SEGMENT, the PART\_VAR is expected to

point to a variable with values of 0 and 1, indicating whether the variable was expected to be observed for each data row. If HIERARCHY, this is also checked recursively, so, if a variable points to such a participation variable, and that other variable does has also a PART\_VAR entry pointing to a variable, the observation of the initial variable is only expected, if both segment variables are 1.

pretty_print	<a href="#">logical</a> deprecated. If you want to have a human readable output, use SummaryData instead of SummaryTable
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.

### Value

a list with:

- SummaryTable: data frame about item missingness per response variable
- SummaryData: data frame about item missingness per response variable formatted for user
- SummaryPlot: ggplot2 heatmap plot, if show\_causes was TRUE
- ReportSummaryTable: data frame underlying SummaryPlot

### ALGORITHM OF THIS IMPLEMENTATION:

- Lists of missing codes and, if applicable, jump codes are selected from the metadata
- The no. of system missings (NA) in each variable is calculated
- The no. of used missing codes is calculated for each variable
- The no. of used jump codes is calculated for each variable
- Two result dataframes (1: on the level of observations, 2: a summary for each variable) are generated
- *OPTIONAL*: if show\_causes is selected, one summary plot for all resp\_vars is provided

### See Also

[Online Documentation](#)

---

com\_qualified\_item\_missingness

*Compute Indicators for Qualified Item Missingness*

---

### Description

[Indicator](#)

**Usage**

```
com_qualified_item_missingness(
  resp_vars,
  study_data,
  label_col = NULL,
  item_level = "item_level",
  expected_observations = c("HIERARCHY", "ALL", "SEGMENT"),
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

`resp_vars` [variable list](#) the name of the measurement variables

`study_data` [data.frame](#) the data frame that contains the measurements

`label_col` [variable attribute](#) the name of the column in the metadata with labels of variables

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`expected_observations` [enum](#) HIERARCHY | ALL | SEGMENT. Report the number of observations expected using the old PART\_VAR concept. See [com\\_item\\_missingness](#) for an explanation.

`meta_data` [data.frame](#) old name for item\_level

`meta_data_v2` [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify meta\_data\_v2.

**Value**

A [list](#) with:

- `SummaryTable`: [data.frame](#) containing data quality checks for "Non-response rate" (PCT\_com\_qum\_nonresp) and "Refusal rate" (PCT\_com\_qum\_refusal) for each response variable in `resp_vars`.
- `SummaryData`: a [data.frame](#) containing data quality checks for "Non-response rate" and "Refusal rate" for a report

---

com\_qualified\_segment\_missingness

*Compute Indicators for Qualified Segment Missingness*

---

**Description**

[Indicator](#)

**Usage**

```
com_qualified_segment_missingness(
  label_col = NULL,
  study_data,
  item_level = "item_level",
  expected_observations = c("HIERARCHY", "ALL", "SEGMENT"),
  meta_data = item_level,
  meta_data_v2,
  meta_data_segment,
  segment_level
)
```

**Arguments**

`label_col` [variable attribute](#) the name of the column in the metadata with labels of variables

`study_data` [data.frame](#) the data frame that contains the measurements

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`expected_observations` [enum](#) HIERARCHY | ALL | SEGMENT. Report the number of observations expected using the old PART\_VAR concept. See [com\\_item\\_missingness](#) for an explanation.

`meta_data` [data.frame](#) old name for item\_level

`meta_data_v2` [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify meta\_data\_v2.

`meta_data_segment` [data.frame](#) Segment level metadata

`segment_level` [data.frame](#) alias for meta\_data\_segment

**Value**

A [list](#) with:

- SegmentTable: [data.frame](#) containing data quality checks for "Non-response rate" (PCT\_com\_qum\_nonresp) and "Refusal rate" (PCT\_com\_qum\_refusal) for each segment.
- SegmentData: a [data.frame](#) containing data quality checks for "Unexpected location" and "Unexpected proportion" per segment for a report

---

com\_segment\_missingness

*Summarizes missingness for individuals in specific segments*

---

## Description

### This implementation can be applied in two use cases::

1. participation in study segments is not recorded by respective variables, e.g. a participant's refusal to attend a specific examination is not recorded.
2. participation in study segments is recorded by respective variables.

Use case (1) will be common in smaller studies. For the calculation of segment missingness it is assumed that study variables are nested in respective segments. This structure must be specified in the static metadata. The R-function identifies all variables within each segment and returns TRUE if all variables within a segment are missing, otherwise FALSE.

Use case (2) assumes a more complex structure of study data and metadata. The study data comprise so-called intro-variables (either TRUE/FALSE or codes for non-participation). The column PART\_VAR in the metadata is filled by variable-IDs indicating for each variable the respective intro-variable. This structure has the benefit that subsequent calculation of item missingness obtains correct denominators for the calculation of missingness rates.

[Descriptor](#)

## Usage

```
com_segment_missingness(
  study_data,
  item_level = "item_level",
  strata_vars = NULL,
  group_vars = NULL,
  label_col,
  threshold_value,
  direction,
  color_gradient_direction,
  expected_observations = c("HIERARCHY", "ALL", "SEGMENT"),
  exclude_roles = c(VARIABLE_ROLES$PROCESS),
  meta_data = item_level,
  meta_data_v2,
  segment_level,
  meta_data_segment
)
```

## Arguments

study_data	<a href="#">data.frame</a> the data frame that contains the measurements
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
strata_vars	<a href="#">variable</a> the name of a variable used for stratification, defaults to NULL for not grouping output
group_vars	<a href="#">variable</a> the name of a variable used for grouping, defaults to <i>NULL</i> for not grouping output
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
threshold_value	<a href="#">numeric</a> from=0 to=100. a numerical value ranging from 0-100

direction	<b>enum</b> low   high. "high" or "low", i.e. are deviations above/below the threshold critical. This argument is deprecated and replaced by <i>color_gradient_direction</i> .
color_gradient_direction	<b>enum</b> above   below. "above" or "below", i.e. are deviations above or below the threshold critical? (default: above)
expected_observations	<b>enum</b> HIERARCHY   ALL   SEGMENT. If ALL, all observations are expected to comprise all study segments. If SEGMENT, the PART_VAR is expected to point to a variable with values of 0 and 1, indicating whether the variable was expected to be observed for each data row. If HIERARCHY, this is also checked recursively, so, if a variable points to such a participation variable, and that other variable does has also a PART_VAR entry pointing to a variable, the observation of the initial variable is only expected, if both segment variables are 1.
exclude_roles	<b>variable roles</b> a character (vector) of variable roles not included
meta_data	<b>data.frame</b> old name for item_level
meta_data_v2	<b>character</b> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
segment_level	<b>data.frame</b> alias for meta_data_segment
meta_data_segment	<b>data.frame</b> Segment level metadata. Optional.

## Details

### Implementation and use of thresholds:

This implementation uses one threshold to discriminate critical from non-critical values. If direction is above than all values below the threshold\_value are normal (displayed in dark blue in the plot and flagged with GRADING = 0 in the dataframe). All values above the threshold\_value are considered critical. The more they deviate from the threshold the displayed color shifts to dark red. All critical values are highlighted with GRADING = 1 in the summary data frame. By default, highest values are always shown in dark red irrespective of the absolute deviation.

If direction is below than all values above the threshold\_value are normal (displayed in dark blue, GRADING = 0).

### Hint:

This function does not support a resp\_vars argument but exclude\_roles to specify variables not relevant for detecting a missing segment.

List function.

## Value

a list with:

- ResultData: data frame about segment missingness
- SummaryPlot: ggplot2 heatmap plot: a heatmap-like graphic that highlights critical values depending on the respective threshold\_value and direction.
- ReportSummaryTable: data frame underlying SummaryPlot

**See Also**

[Online Documentation](#)

---

com\_unit\_missingness *Counts all individuals with no measurements at all*

---

**Description**

This implementation examines a crude version of unit missingness or unit-nonresponse (Kalton and Kasprzyk 1986), i.e. if all measurement variables in the study data are missing for an observation it has unit missingness.

The function can be applied on stratified data. In this case strata\_vars must be specified.

[Descriptor](#)

**Usage**

```
com_unit_missingness(
  id_vars = NULL,
  strata_vars = NULL,
  label_col,
  study_data,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

id_vars	<a href="#">variable list</a> optional, a (vectorized) call of ID-variables that should not be considered in the calculation of unit- missingness
strata_vars	<a href="#">variable</a> optional, a string or integer variable used for stratification
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.

## Details

This implementations calculates a crude rate of unit-missingness. This type of missingness may have several causes and is an important research outcome. For example, unit-nonresponse may be selective regarding the targeted study population or technical reasons such as record-linkage may cause unit-missingness.

It has to be discriminated form segment and item missingness, since different causes and mechanisms may be the reason for unit-missingness.

### Hint:

This function does not support a `resp_vars` argument but `id_vars`, which have a roughly inverse logic behind: `id_vars` with values do not prevent a row from being considered missing, because an ID is the only hint for a unit that otherwise would not occur in the data at all.

List function.

## Value

A list with:

- FlaggedStudyData: [data.frame](#) with id-only-rows flagged in a column `Unit_missing`
- SummaryData: [data.frame](#) with numbers and percentages of unit missingness

## See Also

[Online Documentation](#)

---

contradiction\_functions\_descriptions

*description of the contradiction functions*

---

## Description

description of the contradiction functions

## Usage

```
contradiction_functions_descriptions
```

## Format

An object of class `list` of length 11.

---

CONTRADICTION\_TERM      *Cross-item level metadata attribute name*

---

**Description**

Note: in some prep\_-functions, this field is named RULE

**Usage**

CONTRADICTION\_TERM

**Format**

An object of class character of length 1.

**Details**

Specifies a contradiction rule. Use REDCap like syntax, see [online vignette](#)

**See Also**

[meta\\_data\\_cross](#)

Other meta\_data\_cross: ASSOCIATION\_DIRECTION, ASSOCIATION\_FORM, ASSOCIATION\_METRIC, ASSOCIATION\_RANGE, CHECK\_ID, CHECK\_LABEL, COMPUTED\_VARIABLE\_ROLES, CONTRADICTION\_TYPE, DATA\_PREPARATION, GOLDSTANDARD, IRV, MAHALANOBIS\_THRESHOLD, MAXIMUM\_LONG\_STRING, MISS\_RESP, MULTIVARIATE\_OUTLIER\_CHECK, MULTIVARIATE\_OUTLIER\_CHECKTYPE, RELCOMPL\_SPEED, REL\_VAL, RESPT\_PER\_ITEM, SCALE\_ACRONYM, SCALE\_NAME, TOTRESPT, VARIABLE\_LIST, VARIABLE\_LIST\_ORDER, meta\_data\_computation, meta\_data\_cross

---

CONTRADICTION\_TYPE      *Cross-item level metadata attribute name*

---

**Description**

Specifies the type of a contradiction. According to the data quality concept, there are logical and empirical contradictions, see [online vignette](#)

**Usage**

CONTRADICTION\_TYPE

**Format**

An object of class character of length 1.

**See Also**[meta\\_data\\_cross](#)

Other `meta_data_cross`: [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

---

`con_contradictions`      *Checks user-defined contradictions in study data*

---

**Description**

This approach considers a contradiction if impossible combinations of data are observed in one participant. For example, if age of a participant is recorded repeatedly the value of age is (unfortunately) not able to decline. Most cases of contradictions rest on comparison of two variables.

Important to note, each value that is used for comparison may represent a possible characteristic but the combination of these two values is considered to be impossible. The approach does not consider implausible or inadmissible values.

[Descriptor](#)**Usage**

```
con_contradictions(
  resp_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  threshold_value,
  check_table,
  summarize_categories = FALSE,
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

<code>resp_vars</code>	<a href="#">variable list</a> the name of the measurement variables
<code>study_data</code>	<a href="#">data.frame</a> the data frame that contains the measurements
<code>label_col</code>	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
<code>item_level</code>	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
<code>threshold_value</code>	<a href="#">numeric</a> from=0 to=100. a numerical value ranging from 0-100
<code>check_table</code>	<a href="#">data.frame</a> contradiction rules table. Table defining contradictions. See details for its required structure.

summarize_categories	<a href="#">logical</a> Needs a column 'tag' in the check_table. If set, a summary output is generated for the defined categories plus one plot per category.
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.

## Details

### Algorithm of this implementation::

- Select all variables in the data with defined contradiction rules (static metadata column CONTRADICTIONS)
- Remove missing codes from the study data (if defined in the metadata)
- Remove measurements deviating from limits defined in the metadata
- Assign label to levels of categorical variables (if applicable)
- Apply contradiction checks on predefined sets of variables
- Identification of measurements fulfilling contradiction rules. Therefore two output data frames are generated:
  - on the level of observation to flag each contradictory value combination, and
  - a summary table for each contradiction check.
- A summary plot illustrating the number of contradictions is generated.

List function.

## Value

If summarize\_categories is FALSE: A [list](#) with:

- `FlaggedStudyData`: The first output of the contradiction function is a data frame of similar dimension regarding the number of observations in the study data. In addition, for each applied check on the variables an additional column is added which flags observations with a contradiction given the applied check.
- `SummaryTable`: The second output summarizes this information into one data frame. This output can be used to provide an executive overview on the amount of contradictions. This output is meant for automatic digestion within pipelines.
- `SummaryData`: The third output is the same as `SummaryTable` but for human readers.
- `SummaryPlot`: The fourth output visualizes summarized information of `SummaryData`.

if summarize\_categories is TRUE, other objects are returned: one per category named by that category (e.g. "Empirical") containing a result for contradictions within that category only. Additionally, in the slot `all_checks` a result as it would have been returned with `summarize_categories` set to FALSE. Finally, a slot `SummaryData` is returned containing sums per Category and an according [ggplot2::ggplot](#) in `SummaryPlot`.

## See Also

[Online Documentation](#)

---

con\_contradictions\_redcap

*Checks user-defined contradictions in study data*

---

## Description

This approach considers a contradiction if impossible combinations of data are observed in one participant. For example, if age of a participant is recorded repeatedly the value of age is (unfortunately) not able to decline. Most cases of contradictions rest on comparison of two variables.

Important to note, each value that is used for comparison may represent a possible characteristic but the combination of these two values is considered to be impossible. The approach does not consider implausible or inadmissible values.

[Indicator](#)

## Usage

```
con_contradictions_redcap(
  study_data,
  item_level = "item_level",
  label_col,
  threshold_value,
  meta_data_cross_item = "cross-item_level",
  use_value_labels,
  summarize_categories = FALSE,
  meta_data = item_level,
  cross_item_level,
  `cross-item_level`,
  meta_data_v2
)
```

## Arguments

`study_data` [data.frame](#) the data frame that contains the measurements

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`label_col` [variable attribute](#) the name of the column in the metadata with labels of variables

`threshold_value` [numeric](#) from=0 to=100. a numerical value ranging from 0-100

`meta_data_cross_item` [data.frame](#) contradiction rules table. Table defining contradictions. See [online documentation](#) for its required structure.

`use_value_labels` [logical](#) Deprecated in favor of [DATA\\_PREPARATION](#). If set to TRUE, labels can be used in the REDCap syntax to specify contraction checks for categorical variables. If set to FALSE, contractions have to be specified using the coded values. In case that this argument is not set in the function call, it will be set to TRUE if the metadata contains a column VALUE\_LABELS which is not empty.

`summarize_categories`  
     **logical** Needs a column `CONTRADICTION_TYPE` in the `meta_data_cross_item`. If set, a summary output is generated for the defined categories plus one plot per category. TODO: Not yet controllable by metadata.

`meta_data`      **data.frame** old name for `item_level`

`cross_item_level`  
     **data.frame** alias for `meta_data_cross_item`

`meta_data_v2`    **character** path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

`'cross-item_level'`  
     **data.frame** alias for `meta_data_cross_item`

## Details

### Algorithm of this implementation::

- Remove missing codes from the study data (if defined in the metadata)
- Remove measurements deviating from limits defined in the metadata
- Assign label to levels of categorical variables (if applicable)
- Apply contradiction checks (given as REDCap-like rules in a separate metadata table)
- Identification of measurements fulfilling contradiction rules. Therefore two output data frames are generated:
  - on the level of observation to flag each contradictory value combination, and
  - a summary table for each contradiction check.
- A summary plot illustrating the number of contradictions is generated.

List function.

## Value

If `summarize_categories` is `FALSE`: A [list](#) with:

- `FlaggedStudyData`: The first output of the contradiction function is a data frame of similar dimension regarding the number of observations in the study data. In addition, for each applied check on the variables an additional column is added which flags observations with a contradiction given the applied check.
- `VariableGroupData`: The second output summarizes this information into one data frame. This output can be used to provide an executive overview on the amount of contradictions.
- `VariableGroupTable`: A subset of `VariableGroupData` used within the pipeline.
- `SummaryPlot`: The third output visualizes summarized information of `SummaryData`.

If `summarize_categories` is `TRUE`, other objects are returned: A list with one element `Other`, a list with the following entries: One per category named by that category (e.g. "Empirical") containing a result for contradiction checks within that category only. Additionally, in the slot `all_checks`, a result as it would have been returned with `summarize_categories` set to `FALSE`. Finally, in the top-level list, a slot `SummaryData` is returned containing sums per `Category` and an according [ggplot2::ggplot](#) in `SummaryPlot`.

**See Also**

[Online Documentation for the function meta\\_data\\_cross](#) [Online Documentation for the required cross-item-level metadata](#)

---

con\_inadmissible\_categorical  
*Detects variable levels not specified in metadata*

---

**Description**

For each categorical variable, value lists should be defined in the metadata. This implementation will examine, if all observed levels in the study data are valid.

[Indicator](#)

**Usage**

```
con_inadmissible_categorical(  
  resp_vars = NULL,  
  study_data,  
  label_col,  
  item_level = "item_level",  
  threshold_value = 0,  
  meta_data = item_level,  
  meta_data_v2  
)
```

**Arguments**

resp_vars	<a href="#">variable list</a> the name of the measurement variables
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
threshold_value	<a href="#">numeric</a> from=0 to=100. a numerical value ranging from 0-100.
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.

## Details

### Algorithm of this implementation::

- Remove missing codes from the study data (if defined in the metadata)
- Interpretation of variable specific VALUE\_LABELS as supplied in the metadata.
- Identification of measurements not corresponding to the expected categories. Therefore two output data frames are generated:
  - on the level of observation to flag each undefined category, and
  - a summary table for each variable.
- Values not corresponding to defined categories are removed in a data frame of modified study data

## Value

a list with:

- SummaryData: data frame summarizing inadmissible categories with the columns:
  - Variables: variable name/label
  - OBSERVED\_CATEGORIES: the categories observed in the study data
  - DEFINED\_CATEGORIES: the categories defined in the metadata
  - NON\_MATCHING: the categories observed but not defined
  - NON\_MATCHING\_N: the number of observations with categories not defined
  - NON\_MATCHING\_N\_PER\_CATEGORY: the number of observations for each of the unexpected categories
- SummaryTable: data frame for the dataquieR pipeline reporting the number and percentage of inadmissible categorical values
- ModifiedStudyData: study data having inadmissible categories removed
- FlaggedStudyData: study data having cases with inadmissible categories flagged

## See Also

[Online Documentation](#)

---

con\_inadmissible\_vocabulary

*Detects variable levels not specified in standardized vocabulary*

---

## Description

For each categorical variable, value lists should be defined in the metadata. This implementation will examine, if all observed levels in the study data are valid.

[Indicator](#)

**Usage**

```
con_inadmissible_vocabulary(
  resp_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  threshold_value = 0,
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

**resp\_vars**      [variable list](#) the name of the measurement variables  
**study\_data**     [data.frame](#) the data frame that contains the measurements  
**label\_col**       [variable attribute](#) the name of the column in the metadata with labels of variables  
**item\_level**      [data.frame](#) the data frame that contains metadata attributes of study data  
**threshold\_value**      [numeric](#) from=0 to=100. a numerical value ranging from 0-100.  
**meta\_data**       [data.frame](#) old name for item\_level  
**meta\_data\_v2**    [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify meta\_data\_v2.

**Details****Algorithm of this implementation::**

- Remove missing codes from the study data (if defined in the metadata)
- Interpretation of variable specific VALUE\_LABELS as supplied in the metadata.
- Identification of measurements not corresponding to the expected categories. Therefore two output data frames are generated:
  - on the level of observation to flag each undefined category, and
  - a summary table for each variable.
- Values not corresponding to defined categories are removed in a data frame of modified study data

**Value**

a list with:

- SummaryData: data frame summarizing inadmissible categories with the columns:
  - Variables: variable name/label
  - OBSERVED\_CATEGORIES: the categories observed in the study data
  - DEFINED\_CATEGORIES: the categories defined in the metadata
  - NON\_MATCHING: the categories observed but not defined

- NON\_MATCHING\_N: the number of observations with categories not defined
- NON\_MATCHING\_N\_PER\_CATEGORY: the number of observations for each of the unexpected categories
- GRADING: indicator TRUE/FALSE if inadmissible categorical values were observed (more than indicated by the threshold\_value)
- SummaryTable: data frame for the dataquieR pipeline reporting the number and percentage of inadmissible categorical values
- ModifiedStudyData: study data having inadmissible categories removed
- FlaggedStudyData: study data having cases with inadmissible categories flagged

## See Also

[Online Documentation](#)

## Examples

```
## Not run:
sdt <- data.frame(DIAG = c("B050", "B051", "B052", "B999"),
                 MED0 = c("S01XA28", "N07XX18", "ABC", NA), stringsAsFactors = FALSE)
mdt <- tibble::tribble(
  ~ VAR_NAMES, ~ DATA_TYPE, ~ STANDARDIZED_VOCABULARY_TABLE, ~ SCALE_LEVEL, ~ LABEL,
  "DIAG", "string", "<ICD10>", "nominal", "Diagnosis",
  "MED0", "string", "<ATC>", "nominal", "Medication"
)
con_inadmissible_vocabulary(NULL, sdt, mdt, label_col = LABEL)
prep_load_workbook_like_file("meta_data_v2")
il <- prep_get_data_frame("item_level")
il$STANDARDIZED_VOCABULARY_TABLE[[11]] <- "<ICD10GM>"
il$DATA_TYPE[[11]] <- DATA_TYPES$INTEGER
il$SCALE_LEVEL[[11]] <- SCALE_LEVELS$NOMINAL
prep_add_data_frames(item_level = il)
r <- dq_report2("study_data", dimensions = "con")
r <- dq_report2("study_data", dimensions = "con",
               advanced_options = list(dataquieR.non_disclosure = TRUE))
r

## End(Not run)
```

---

con\_limit\_deviations *Detects variable values exceeding limits defined in metadata*

---

## Description

Inadmissible numerical values can be of type integer or float. This implementation requires the definition of intervals in the metadata to examine the admissibility of numerical study data.

This helps identify inadmissible measurements according to hard limits (for multiple variables).

[Indicator](#)

**Usage**

```

con_limit_deviations(
  resp_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  meta_data_cross_item = "cross-item_level",
  limits = NULL,
  flip_mode = "noflip",
  return_flagged_study_data = FALSE,
  return_limit_categorical = TRUE,
  meta_data = item_level,
  cross_item_level,
  `cross-item_level`,
  meta_data_v2,
  show_obs = TRUE
)

```

**Arguments**

**resp\_vars** [variable list](#) the name of the measurement variables  
**study\_data** [data.frame](#) the data frame that contains the measurements  
**label\_col** [variable attribute](#) the name of the column in the metadata with labels of variables  
**item\_level** [data.frame](#) the data frame that contains metadata attributes of study data  
**meta\_data\_cross\_item** [meta\\_data\\_cross](#)  
**limits** [enum](#) HARD\_LIMITS | SOFT\_LIMITS | DETECTION\_LIMITS. what limits from metadata to check for  
**flip\_mode** [enum](#) default | flip | noflip | auto. Should the plot be in default orientation, flipped, not flipped or auto-flipped. Not all options are always supported. In general, this can be controlled by setting the `roptions(dataquieR.flip_mode = ...)`. If called from `dq_report`, you can also pass `flip_mode` to all function calls or set them specifically using `specific_args`.  
**return\_flagged\_study\_data** [logical](#) return `FlaggedStudyData` in the result  
**return\_limit\_categorical** [logical](#) if TRUE return limit deviations also for categorical variables  
**meta\_data** [data.frame](#) old name for `item_level`  
**cross\_item\_level** [data.frame](#) alias for `meta_data_cross_item`  
**meta\_data\_v2** [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.  
**show\_obs** [logical](#) Should (selected) individual observations be marked in the figure for continuous variables?  
**`cross-item\_level`** [data.frame](#) alias for `meta_data_cross_item`

## Details

### Algorithm of this implementation::

- Remove missing codes from the study data (if defined in the metadata)
- Interpretation of variable specific intervals as supplied in the metadata.
- Identification of measurements outside defined limits. Therefore two output data frames are generated:
  - on the level of observation to flag each deviation, and
  - a summary table for each variable.
- A list of plots is generated for each variable examined for limit deviations. The histogram-like plots indicate respective limits as well as deviations.
- Values exceeding limits are removed in a data frame of modified study data

## Value

a list with:

- FlaggedStudyData [data.frame](#) related to the study data by a 1:1 relationship, i.e. for each observation is checked whether the value is below or above the limits. Optional, see `return_flagged_study_data`.
- SummaryTable [data.frame](#) summarizing limit deviations for each variable.
- SummaryData [data.frame](#) summarizing limit deviations for each variable for a report.
- SummaryPlotList [list](#) of [ggplot2::ggplots](#) The plots for each variable are either a histogram (continuous) or a barplot (discrete).
- ReportSummaryTable: heatmap-like data frame about limit violations

## See Also

- [Online Documentation](#)

---

dataquieR.acc\_loess.exclude\_constant\_subgroups

*Exclude subgroups with constant values from LOESS figure*

---

## Description

If this option is set to TRUE, time course plots will only show subgroups with more than one distinct value. This might improve the readability of the figure.

## See Also

Other options: [dataquieR](#), [dataquieR.CONDITIONS\\_LEVEL\\_TRHESHOLD](#), [dataquieR.CONDITIONS\\_WITH\\_STACKTRACE](#), [dataquieR.ELEMENT\\_MISSMATCH\\_CHECKTYPE](#), [dataquieR.ERRORS\\_WITH\\_CALLER](#), [dataquieR.GAM\\_for\\_LOESS](#), [dataquieR.MAHALANOBIS\\_THRESHOLD](#), [dataquieR.MAX\\_LABEL\\_LEN](#), [dataquieR.MAX\\_LONG\\_LABEL\\_LEN](#), [dataquieR.MAX\\_VALUE\\_LABEL\\_LEN](#), [dataquieR.MESSAGES\\_WITH\\_CALLER](#), [dataquieR.MULTIVARIATE\\_OUTLIER\\_CHECK](#), [dataquieR.VALUE\\_LABELS\\_htmlescaped](#), [dataquieR.WARNINGS\\_WITH\\_CALLER](#), [dataquieR.acc\\_loess.mark\\_time\\_point](#), [dataquieR.acc\\_loess.min\\_bw](#), [dataquieR.acc\\_loess.min\\_proportion](#), [dataquieR.acc\\_loess.plot\\_format](#),

dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort,  
 dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logi  
 dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove,  
 dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust,  
 dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes,  
 dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets,  
 dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars,  
 dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility,  
 dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot,  
 dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot,  
 dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling,  
 dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor,  
 dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint,  
 dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit,  
 dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max,  
 dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache  
 dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback,  
 dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.acc\_loess.mark\_time\_points

*Display time-points in LOESS plots*

---

## Description

TODO

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE,  
 dataquieR.ELEMENT\_MISSMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS,  
 dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN,  
 dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK,  
 dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const  
 dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format,  
 dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort,  
 dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logi  
 dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove,  
 dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust,  
 dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes,  
 dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets,  
 dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars,  
 dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility,  
 dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot,  
 dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot,  
 dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling,  
 dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor,

dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.acc\_loess.min\_bw

*Lower limit for the LOESS bandwidth*

---

### Description

The value should be greater than 0 and less than or equal to 1. In general, increasing the bandwidth leads to a smoother trend line.

### See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_for, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logi, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.acc\_loess.min\_proportion

*Lower limit for the proportion of cases or controls to create a smoothed time trend figure*

---

### Description

The value should be greater than 0 and less than 0.4. If the proportion of cases or controls is lower than the specified value, the LOESS figure will not be created for the specified binary outcome.

### See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logi`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.acc\_loess.plot\_format

*default for Plot-Format in acc\_loess()*

---

### Description

TODO

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logi`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecode`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

`dataquieR.acc_loess.plot_observations`

*Display observations in LOESS plots*

---

**Description**

TODO

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logi`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`,

dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.acc\_margins\_num

*Include number of observations for each level of the grouping variable in the 'margins' figure*

---

### Description

If this option is set to FALSE, the figures created by acc\_margins will not include the number of observations for each level of the grouping variable. This can be used to obtain clean static plots.

### See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_THRESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logi, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit,

dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.acc\_margins\_sort

*Sort levels of the grouping variable in the 'margins' figures*

---

### Description

If this option is set to TRUE, the levels of the grouping variable in the figure are sorted in descending order according to the number of observations so that levels with more observations are easier to identify. Otherwise, the original order of the levels is retained.

### See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logi, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.acc\_multivariate\_outlier.scale

*Apply min-max scaling in parallel coordinates figure to inspect multivariate outliers*

---

### Description

boolean, TRUE or FALSE

### See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.applicability\_problem

*An exception class assigned for exceptions caused by trying to apply a non-applicable indicator function*

---

### Description

Amending metadata could make the function running, e.g., a test for missingness without any declared missing codes

**Usage**

```
dataquieR.applicability_problem
```

**Format**

An object of class character of length 1.

---

```
dataquieR.col_con_con_empirical
```

*Color for empirical contradictions*

---

**Description**

TODO

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.col\_con\_con\_logical

*Color for logical contradictions*

---

### Description

TODO

### See Also

Other options: [dataquieR](#), [dataquieR.CONDITIONS\\_LEVEL\\_TRHESHOLD](#), [dataquieR.CONDITIONS\\_WITH\\_STACKTRACE](#), [dataquieR.ELEMENT\\_MISMATCH\\_CHECKTYPE](#), [dataquieR.ERRORS\\_WITH\\_CALLER](#), [dataquieR.GAM\\_for\\_LOESS](#), [dataquieR.MAHALANOBIS\\_THRESHOLD](#), [dataquieR.MAX\\_LABEL\\_LEN](#), [dataquieR.MAX\\_LONG\\_LABEL\\_LEN](#), [dataquieR.MAX\\_VALUE\\_LABEL\\_LEN](#), [dataquieR.MESSAGES\\_WITH\\_CALLER](#), [dataquieR.MULTIVARIATE\\_OUTLIER\\_CHECK](#), [dataquieR.VALUE\\_LABELS\\_htmlescaped](#), [dataquieR.WARNINGS\\_WITH\\_CALLER](#), [dataquieR.acc\\_loess.exclude\\_const](#), [dataquieR.acc\\_loess.mark\\_time\\_points](#), [dataquieR.acc\\_loess.min\\_bw](#), [dataquieR.acc\\_loess.min\\_proportion](#), [dataquieR.acc\\_loess.plot\\_format](#), [dataquieR.acc\\_loess.plot\\_observations](#), [dataquieR.acc\\_margins\\_num](#), [dataquieR.acc\\_margins\\_sort](#), [dataquieR.acc\\_multivariate\\_outlier.scale](#), [dataquieR.col\\_con\\_con\\_empirical](#), [dataquieR.convert\\_to\\_list\\_for\\_lapply](#), [dataquieR.debug](#), [dataquieR.des\\_summary\\_hard\\_lim\\_remove](#), [dataquieR.dontwrapresults](#), [dataquieR.droplevels\\_ReportSummaryTable](#), [dataquieR.dt\\_adjust](#), [dataquieR.fix\\_column\\_type\\_on\\_read](#), [dataquieR.flip\\_mode](#), [dataquieR.force\\_item\\_specific\\_missing\\_codes](#), [dataquieR.force\\_label\\_col](#), [dataquieR.grading\\_formats](#), [dataquieR.grading\\_rulesets](#), [dataquieR.guess\\_character](#), [dataquieR.guess\\_missing\\_codes](#), [dataquieR.ignore\\_empty\\_vars](#), [dataquieR.lang](#), [dataquieR.lazy\\_plots](#), [dataquieR.lazy\\_plots\\_cache](#), [dataquieR.lazy\\_plots\\_gg\\_compatibility](#), [dataquieR.locale](#), [dataquieR.max\\_cat\\_resp\\_var\\_levels\\_in\\_plot](#), [dataquieR.max\\_group\\_var\\_levels\\_in\\_plot](#), [dataquieR.max\\_group\\_var\\_levels\\_with\\_violins](#), [dataquieR.min\\_obs\\_per\\_group\\_var\\_in\\_plot](#), [dataquieR.min\\_time\\_points\\_for\\_cat\\_resp\\_var](#), [dataquieR.non\\_disclosure](#), [dataquieR.old\\_factor\\_handling](#), [dataquieR.old\\_type\\_adjust](#), [dataquieR.precomputeStudyData](#), [dataquieR.print\\_block\\_load\\_factor](#), [dataquieR.progress\\_fkt\\_default](#), [dataquieR.progress\\_msg\\_fkt\\_default](#), [dataquieR.resume\\_checkpoint](#), [dataquieR.resume\\_print](#), [dataquieR.scale\\_level\\_heuristics\\_control\\_binaryrecodelimit](#), [dataquieR.scale\\_level\\_heuristics\\_control\\_metriclevels](#), [dataquieR.study\\_data\\_cache\\_max](#), [dataquieR.study\\_data\\_cache\\_metrics](#), [dataquieR.study\\_data\\_cache\\_metrics\\_env](#), [dataquieR.study\\_data\\_cache](#), [dataquieR.study\\_data\\_colnames\\_case\\_sensitive](#), [dataquieR.testdebug](#), [dataquieR.traceback](#), [dataquieR.type\\_adjust\\_parallel](#), [dataquieR.progress\\_init\\_fkt](#)

---

dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD

*Log Level*

---

### Description

TODO

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_html_escaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_constant_subgroups`, `dataquieR.acc_loess.max_obs_per_group_var`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_loglik`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodeLimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache_max_obs_per_group_var`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

`dataquieR.CONDITIONS_WITH_STACKTRACE`

*Add stack-trace in condition messages (to be deprecated)*

---

**Description**

to be deprecated

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_THRESHOLD`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_html_escaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_constant_subgroups`, `dataquieR.acc_loess.max_obs_per_group_var`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_loglik`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`,

dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.convert\_to\_list\_for\_lapply

*If report uses a storr back-end, do not convert to base-list*

---

### Description

if TRUE and a report uses a storr-back-end, convert it to a base list, i.e., copy to the RAM, even if this would likely not be really needed for apply-calls

### See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISSMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max,

dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_metrics\_max,  
 dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback,  
 dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.debug	<i>Call <code>browser()</code> on errors</i>
-----------------	--

---

## Description

TODO

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_THRESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_constants, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.des\_summary\_hard\_limit\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_metrics\_max, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.des_summary_hard_lim_remove	<i>Removal of hard limits from data before calculating descriptive statistics.</i>
---------------------------------------	--

---

**Description**

can be

- TRUE: values outside hard limits will be removed from the data before calculating descriptive statistics
- FALSE: values outside hard limits will not be removed from the original data

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

`dataquieR.dontwrapresults`

*Disable automatic post-processing of dataquieR function results*

---

**Description**

TODO

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`,

dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK,  
 dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const,  
 dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion,  
 dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num,  
 dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical,  
 dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug,  
 dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust,  
 dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes,  
 dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets,  
 dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars,  
 dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility,  
 dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot,  
 dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot,  
 dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling,  
 dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor,  
 dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint,  
 dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodeLimit,  
 dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max,  
 dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache,  
 dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback,  
 dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.droplevels\_ReportSummaryTable

*Show also unused levels in heatmaps*

---

## Description

if TRUE, levels not taken will not be displayed when printing/plotting heatmap tables

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE,  
 dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS,  
 dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN,  
 dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK,  
 dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const,  
 dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion,  
 dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num,  
 dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical,  
 dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug,  
 dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.dt\_adjust,  
 dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes,  
 dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets,  
 dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars,  
 dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility,  
 dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot,

dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.dt\_adjust     *character* Adjust data types according to metadata

---

## Description

also reports inadmissible data types. can be turned off for performance reasons, if the data source is already type-safe (e.g., a database) **use with care, may cause pipelines breaking (maybe only in the final rendering step), if the data type is incorrectly set for some columns.**

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.ELEMENT\_MISSMATCH\_CHECKTYPE

*Metadata describes more than the current study data*

---

## Description

- none: no check will be provided about the match of variables and records available in the study data and described in the metadata
- exact: There must be a 1:1 match between the study data and metadata regarding data frames and segments variables and records
- subset\_u: study data are a subset of metadata. All variables from the study data are expected to be present in the metadata, but one or more variables in the metadata are not expected to be present in the study data. In this case a variable present in the study data but not in the metadata would produce an issue.
- subset\_m: metadata are a subset of study data. All variables in the metadata are expected to be present in the study data, but one or more variables in the study data are not expected to be present in the metadata.

## See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlspecialchars`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_constant_subgroups`, `dataquieR.acc_loess.max_obs_per_group_var_in_plot`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logit`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache_max_obs_per_group_var_in_plot`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.ERRORS\_WITH\_CALLER

*Set caller for error conditions (to be deprecated)*

---

### Description

to be deprecated

### See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlspecialchars`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_constant_subgroups`, `dataquieR.acc_loess.max`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logit`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.fix\_column\_type\_on\_read

*Try to avoid fallback to string columns when reading files*

---

### Description

If a file does not feature column data types or features data types cell-based, choose that type which matches the majority of the sampled cells of a column for the column's data type.

**Details**

This may make you miss data type problems but it could fix them, so `prep_get_data_frame()` works better.

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

`dataquieR.flip_mode`     *Flip-Mode to Use for figures*

---

**Description**

TODO

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`,

dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical,  
 dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug,  
 dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT,  
 dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.force\_item\_specific\_missing\_codes,  
 dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets,  
 dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars,  
 dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility,  
 dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot,  
 dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot,  
 dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling,  
 dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor,  
 dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint,  
 dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit,  
 dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max,  
 dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache,  
 dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback,  
 dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.force\_item\_specific\_missing\_codes

*Converting **MISSING\_LIST/JUMP\_LIST** to a **MISSING\_LIST\_TABLE**  
create on list per item*

---

## Description

TODO

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE,  
 dataquieR.ELEMENT\_MISSMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS,  
 dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN,  
 dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK,  
 dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const,  
 dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion,  
 dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num,  
 dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical,  
 dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug,  
 dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT,  
 dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_label\_col,  
 dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character,  
 dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots,  
 dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale,  
 dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot,  
 dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot,  
 dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling,  
 dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor,

dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodeLimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.force\_label\_col

*Control, how the label\_col argument is used.*

---

## Description

TODO

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_THRESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_escaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_constants, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specification, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodeLimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.GAM\_for\_LOESS

*Enable to switch to a general additive model instead of LOESS*

---

### Description

If this option is set to TRUE, time course plots will use general additive models (GAM) instead of LOESS when the number of observations exceeds a specified threshold. LOESS computations for large datasets have a high memory consumption.

### See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlspecialchars`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_constant_subgroups`, `dataquieR.acc_loess.max_obs_per_group_var_in_plot`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_loglik`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache_max_obs_per_group_var_in_plot`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.grading\_formats

*Name of the data.frame featuring a format for grading-values*

---

### Description

TODO

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodeLimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

`dataquieR.grading_rulesets`

*Name of the data.frame featuring GRADING\_RULESET*

---

**Description**

TODO

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`

dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metric\_names, dataquieR.study\_data\_cache\_quick\_fill, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.guess\_character

*For metadata guessing, try to guess **DATA\_TYPE** from the data values*

---

## Description

By default, the **DATA\_TYPE** is derived from the R data type of the study data. However, when data are imported from plain text files, it can be more appropriate to examine the actual values and infer the data type based on their content. This option enables that behavior: set `dataquieR.guess_character` to `TRUE` to infer data types from the observed values rather than relying solely on the column's class in the data frame.

## See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_constant`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`,

dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodeLimit,  
 dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max,  
 dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_max,  
 dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback,  
 dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.guess\_missing\_codes

*Control, if dataquieR tries to guess missing-codes from the study data  
 in absence of metadata*

---

## Description

TODO

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_THRESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE,  
 dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS,  
 dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN,  
 dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK,  
 dataquieR.VALUE\_LABELS\_htmlEscaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_constants,  
 dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion,  
 dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num,  
 dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical,  
 dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug,  
 dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable,  
 dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specification,  
 dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets,  
 dataquieR.guess\_character, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots,  
 dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale,  
 dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot,  
 dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot,  
 dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling,  
 dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor,  
 dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint,  
 dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodeLimit,  
 dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max,  
 dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_max,  
 dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback,  
 dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.ignore\_empty\_vars

*character* remove variables with only empty values

---

### Description

remove variables with only empty values (NA, ". ", "" or similar) from reports. auto means, such variables are removed, if we have more than 20% of the variables empty.

### See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

Other `study_data_cache`: `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache_metrics_env_default`, `dataquieR.study_data_cache_quick_fill`

---

dataquieR.intrinsic\_applicability\_problem

*An exception class assigned for exceptions caused by trying to apply a non-applicable indicator function, which is not caused by deficient metadata*

---

**Description**

Also amending meta data could not make the function running, e.g., a test for numbers applied to a character.

**Usage**

```
dataquieR.intrinsic_applicability_problem
```

**Format**

An object of class character of length 1.

---

dataquieR.lang

*Language-Suffix for metadata Label-Columns*

---

**Description**

TODO

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.lazy\_plots *character plots realized lazy*

---

### Description

if TRUE, plots are not realized until needed in side reports to save memory.

### See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISSMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodeLimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.lazy\_plots\_cache  
*character cache realizations*

---

### Description

if TRUE, realized plots are cached, may need more memory.

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodeLimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

`dataquieR.lazy_plots_gg_compatibility`

*character* be as compatible with `ggplot2` objects as possible

---

**Description**

if TRUE, plot promises are blessed in an S7 class so they behave almost like "real" `ggplot2` objects, so you normally do not need to call `prep_realize_ggplot()` on them. However, this comes with a small memory overhead, so, you can disable this.

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`.

dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.locale	<i>character default language for type conversion</i>
------------------	---

---

### Description

the language to use for type conversions (en, de, fr, cn, ca, ...) only used by util\_adjust\_data\_type2(), currently

### See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISSMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit,

dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.MAHALANOBIS\_THRESHOLD

*Default availability of Mahalanobis based multivariate outlier checks in reports*

---

### Description

a number, see corresponding argument in `acc_mahalanobis()`

### See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_THRESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlspecialchars, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_constant\_subgroups, dataquieR.acc\_loess.max, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logit, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.max\_cat\_resp\_var\_levels\_in\_plot

*Maximum number of levels of the categorical response variable shown individually in figures*

---

### Description

If there are more levels of a categorical response variable than can be shown individually, they will be collapsed into "other".

### See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_THRESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metric`, `dataquieR.study_data_cache_quick_fill`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.max\_group\_var\_levels\_in\_plot

*Maximum number of levels of the grouping variable shown individually in figures*

---

### Description

If there are more examiners or devices than can be shown individually, they will be collapsed into "other".

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_viol`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodeLimit`, `dataquieR.scale_level_heuristics_control`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metric`, `dataquieR.study_data_cache_quick_fill`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `dataquieR.progress_init_fkt`

---

`dataquieR.max_group_var_levels_with_violins`

*Maximum number of levels of the grouping variable shown with individual histograms ('violins') in 'margins' figures*

---

**Description**

If there are more examiners or devices, the figure will be reduced to box-plots to save space.

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`

dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metric, dataquieR.study\_data\_cache\_quick\_fill, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.MAX\_LABEL\_LEN

*Maximum length for variable labels LABEL*

---

### Description

All variable labels will be shortened to fit this maximum length. Cannot be larger than 200 for technical reasons.

### See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlspecialchars, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_constant\_subgroups, dataquieR.acc\_loess.max, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logi, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache

[dataquieR.study\\_data\\_colnames\\_case\\_sensitive](#), [dataquieR.testdebug](#), [dataquieR.traceback](#), [dataquieR.type\\_adjust\\_parallel](#), [progress\\_init\\_fkt](#)

---

`dataquieR.MAX_LONG_LABEL_LEN`

*Maximum length for long variable labels [LONG\\_LABEL](#)*

---

### Description

All long variable labels will be shortened to fit this maximum length. Cannot be larger than 200 for technical reasons.

### See Also

Other options: [dataquieR](#), [dataquieR.CONDITIONS\\_LEVEL\\_TRHESHOLD](#), [dataquieR.CONDITIONS\\_WITH\\_STACKTRACE](#), [dataquieR.ELEMENT\\_MISMATCH\\_CHECKTYPE](#), [dataquieR.ERRORS\\_WITH\\_CALLER](#), [dataquieR.GAM\\_for\\_LOESS](#), [dataquieR.MAHALANOBIS\\_THRESHOLD](#), [dataquieR.MAX\\_LABEL\\_LEN](#), [dataquieR.MAX\\_VALUE\\_LABEL\\_LEN](#), [dataquieR.MESSAGES\\_WITH\\_CALLER](#), [dataquieR.MULTIVARIATE\\_OUTLIER\\_CHECK](#), [dataquieR.VALUE\\_LABELS\\_htmlspecialchars](#), [dataquieR.WARNINGS\\_WITH\\_CALLER](#), [dataquieR.acc\\_loess.exclude\\_constant\\_subgroups](#), [dataquieR.acc\\_loess.max\\_colspan](#), [dataquieR.acc\\_loess.min\\_bw](#), [dataquieR.acc\\_loess.min\\_proportion](#), [dataquieR.acc\\_loess.plot\\_format](#), [dataquieR.acc\\_loess.plot\\_observations](#), [dataquieR.acc\\_margins\\_num](#), [dataquieR.acc\\_margins\\_sort](#), [dataquieR.acc\\_multivariate\\_outlier.scale](#), [dataquieR.col\\_con\\_con\\_empirical](#), [dataquieR.col\\_con\\_con\\_loglik](#), [dataquieR.convert\\_to\\_list\\_for\\_lapply](#), [dataquieR.debug](#), [dataquieR.des\\_summary\\_hard\\_lim\\_remove](#), [dataquieR.dontwrapresults](#), [dataquieR.droplevels\\_ReportSummaryTable](#), [dataquieR.dt\\_adjust](#), [dataquieR.fix\\_column\\_type\\_on\\_read](#), [dataquieR.flip\\_mode](#), [dataquieR.force\\_item\\_specific\\_missing\\_codes](#), [dataquieR.force\\_label\\_col](#), [dataquieR.grading\\_formats](#), [dataquieR.grading\\_rulesets](#), [dataquieR.guess\\_character](#), [dataquieR.guess\\_missing\\_codes](#), [dataquieR.ignore\\_empty\\_vars](#), [dataquieR.lang](#), [dataquieR.lazy\\_plots](#), [dataquieR.lazy\\_plots\\_cache](#), [dataquieR.lazy\\_plots\\_gg\\_compatibility](#), [dataquieR.locale](#), [dataquieR.max\\_cat\\_resp\\_var\\_levels\\_in\\_plot](#), [dataquieR.max\\_group\\_var\\_levels\\_in\\_plot](#), [dataquieR.max\\_group\\_var\\_levels\\_with\\_violins](#), [dataquieR.min\\_obs\\_per\\_group\\_var\\_in\\_plot](#), [dataquieR.min\\_time\\_points\\_for\\_cat\\_resp\\_var](#), [dataquieR.non\\_disclosure](#), [dataquieR.old\\_factor\\_handling](#), [dataquieR.old\\_type\\_adjust](#), [dataquieR.precomputeStudyData](#), [dataquieR.print\\_block\\_load\\_factor](#), [dataquieR.progress\\_fkt\\_default](#), [dataquieR.progress\\_msg\\_fkt\\_default](#), [dataquieR.resume\\_checkpoint](#), [dataquieR.resume\\_print](#), [dataquieR.scale\\_level\\_heuristics\\_control\\_binaryrecodelimit](#), [dataquieR.scale\\_level\\_heuristics\\_control\\_metriclevels](#), [dataquieR.study\\_data\\_cache\\_max](#), [dataquieR.study\\_data\\_cache\\_metrics](#), [dataquieR.study\\_data\\_cache\\_metrics\\_env](#), [dataquieR.study\\_data\\_cache\\_max\\_colspan](#), [dataquieR.study\\_data\\_colnames\\_case\\_sensitive](#), [dataquieR.testdebug](#), [dataquieR.traceback](#), [dataquieR.type\\_adjust\\_parallel](#), [progress\\_init\\_fkt](#)

---

`dataquieR.MAX_VALUE_LABEL_LEN`

*Maximum length for value labels*

---

### Description

value labels are restricted to this length

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlspecialchars`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_constant_subgroups`, `dataquieR.acc_loess.max`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logi`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecode`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

`dataquieR.MESSAGES_WITH_CALLER`

*Set caller for message conditions (to be deprecated)*

---

**Description**

to be deprecated

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlspecialchars`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_constant_subgroups`, `dataquieR.acc_loess.max`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logi`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`,

dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.min\_obs\_per\_group\_var\_in\_plot

*Minimum number of observations per grouping variable that is required to include an individual level of the grouping variable in a figure*

---

## Description

Levels of the grouping variable with fewer observations than specified here will be excluded from the figure.

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print,

dataquieR.scale\_level\_heuristics\_control\_binaryrecodeLimit, dataquieR.scale\_level\_heuristics\_control, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metric, dataquieR.study\_data\_cache\_quick\_fill, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.min\_time\_points\_for\_cat\_resp\_var

*Minimum number of data points to create a time course plot for an individual level of a categorical response variable*

---

### Description

If there are less observations for an individual level of a categorical variable, it will not be shown in the time course plot.

### See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_THRESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlEscaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_constant, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodeLimit, dataquieR.scale\_level\_heuristics\_control, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metric, dataquieR.study\_data\_cache\_quick\_fill, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

 dataquieR.MULTIVARIATE\_OUTLIER\_CHECK

*Default availability of multivariate outlier checks in reports*


---

## Description

can be

- TRUE: for cross-item\_level-groups with MULTIVARIATE\_OUTLIER\_CHECK empty, do a multivariate outlier check
- FALSE: for cross-item\_level-groups with MULTIVARIATE\_OUTLIER\_CHECK empty, don't do a multivariate outlier check
- "auto": for cross-item\_level-groups with MULTIVARIATE\_OUTLIER\_CHECK empty, do multivariate outlier checks, if there is no entry in the column [CONTRADICTION\\_TERM](#).

## See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_constant_subgroups`, `dataquieR.acc_loess.m`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logi`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryTable`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_specific_missing_codes`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryreodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.non\_disclosure

*Remove all observation-level-real-data from reports*

---

## Description

TODO

## See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecode`, `dataquieR.scale_level_heuristics_control`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metric`, `dataquieR.study_data_cache_quick_fill`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.old\_factor\_handling

*character use the old handling of study data already featuring factors*

---

## Description

if `study_data` comes as a data frame, it may already feature factors. if a column has the `DATA_TYPE` integer in the meta data, the factor was converted to integer using `as.integer()`, which caused unexpected behavior. if this option is set to "FALSE" (the new default), the conversion will now try to apply `as.character(column_data)`, first.

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodeLimit`, `dataquieR.scale_level_heuristics_control`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metric`, `dataquieR.study_data_cache_quick_fill`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `dataquieR.progress_init_fkt`

---

`dataquieR.old_type_adjust`

*character use the old type conversion code (slower)*

---

**Description**

`character` use the old type conversion code (slower)

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`

dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metric, dataquieR.study\_data\_cache\_quick\_fill, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.precomputeStudyData

*Pre-compute different curation levels of study data*

---

## Description

as described in [dataquieR.study\\_data\\_cache\\_max](#), different flavors of the study data are cached. With this option, you control, if before a report is computed, a frequently needed bunch of such flavors are pre-computed and distributed to the compute nodes. However, this may be time- and RAM-consuming, so, you can turn the pre-computation off, which will still allow the individual compute nodes to maintain such a cache but then growing on demand on individual nodes separately, only. If [dataquieR.study\\_data\\_cache\\_max](#) cannot handle all flavors, they may still be pre-computed but immediately discarded.

## See Also

Other options: [dataquieR](#), [dataquieR.CONDITIONS\\_LEVEL\\_TRHESHOLD](#), [dataquieR.CONDITIONS\\_WITH\\_STACKTRACE](#), [dataquieR.ELEMENT\\_MISSMATCH\\_CHECKTYPE](#), [dataquieR.ERRORS\\_WITH\\_CALLER](#), [dataquieR.GAM\\_for\\_LOESS](#), [dataquieR.MAHALANOBIS\\_THRESHOLD](#), [dataquieR.MAX\\_LABEL\\_LEN](#), [dataquieR.MAX\\_LONG\\_LABEL\\_LEN](#), [dataquieR.MAX\\_VALUE\\_LABEL\\_LEN](#), [dataquieR.MESSAGES\\_WITH\\_CALLER](#), [dataquieR.MULTIVARIATE\\_OUTLIER\\_CHECK](#), [dataquieR.VALUE\\_LABELS\\_htmlescaped](#), [dataquieR.WARNINGS\\_WITH\\_CALLER](#), [dataquieR.acc\\_loess.exclude\\_const](#), [dataquieR.acc\\_loess.mark\\_time\\_points](#), [dataquieR.acc\\_loess.min\\_bw](#), [dataquieR.acc\\_loess.min\\_proportion](#), [dataquieR.acc\\_loess.plot\\_format](#), [dataquieR.acc\\_loess.plot\\_observations](#), [dataquieR.acc\\_margins\\_num](#), [dataquieR.acc\\_margins\\_sort](#), [dataquieR.acc\\_multivariate\\_outlier.scale](#), [dataquieR.col\\_con\\_con\\_empirical](#), [dataquieR.col\\_con\\_con\\_logical](#), [dataquieR.convert\\_to\\_list\\_for\\_lapply](#), [dataquieR.debug](#), [dataquieR.des\\_summary\\_hard\\_lim\\_remove](#), [dataquieR.dontwrapresults](#), [dataquieR.droplevels\\_ReportSummaryT](#), [dataquieR.dt\\_adjust](#), [dataquieR.fix\\_column\\_type\\_on\\_read](#), [dataquieR.flip\\_mode](#), [dataquieR.force\\_item\\_spec](#), [dataquieR.force\\_label\\_col](#), [dataquieR.grading\\_formats](#), [dataquieR.grading\\_rulesets](#), [dataquieR.guess\\_character](#), [dataquieR.guess\\_missing\\_codes](#), [dataquieR.ignore\\_empty\\_vars](#), [dataquieR.lang](#), [dataquieR.lazy\\_plots](#), [dataquieR.lazy\\_plots\\_cache](#), [dataquieR.lazy\\_plots\\_gg\\_compatibility](#), [dataquieR.locale](#), [dataquieR.max\\_cat\\_resp\\_var\\_levels\\_in\\_plot](#), [dataquieR.max\\_group\\_var\\_levels\\_in\\_plot](#), [dataquieR.max\\_group\\_var\\_levels\\_with\\_violins](#), [dataquieR.min\\_obs\\_per\\_group\\_var\\_in\\_plot](#), [dataquieR.min\\_time\\_points\\_for\\_cat\\_resp\\_var](#), [dataquieR.non\\_disclosure](#), [dataquieR.old\\_factor\\_handling](#), [dataquieR.old\\_type\\_adjust](#), [dataquieR.print\\_block\\_load\\_factor](#), [dataquieR.progress\\_fkt\\_default](#),

[dataquieR.progress\\_msg\\_fkt\\_default](#), [dataquieR.resume\\_checkpoint](#), [dataquieR.resume\\_print](#),  
[dataquieR.scale\\_level\\_heuristics\\_control\\_binaryrecodeLimit](#), [dataquieR.scale\\_level\\_heuristics\\_control](#),  
[dataquieR.study\\_data\\_cache\\_max](#), [dataquieR.study\\_data\\_cache\\_metrics](#), [dataquieR.study\\_data\\_cache\\_metrics](#),  
[dataquieR.study\\_data\\_cache\\_quick\\_fill](#), [dataquieR.study\\_data\\_colnames\\_case\\_sensitive](#),  
[dataquieR.testdebug](#), [dataquieR.traceback](#), [dataquieR.type\\_adjust\\_parallel](#), [progress\\_init\\_fkt](#)  
 Other `study_data_cache`: [dataquieR.ignore\\_empty\\_vars](#), [dataquieR.print\\_block\\_load\\_factor](#),  
[dataquieR.study\\_data\\_cache\\_max](#), [dataquieR.study\\_data\\_cache\\_metrics](#), [dataquieR.study\\_data\\_cache\\_metrics](#),  
[dataquieR.study\\_data\\_cache\\_metrics\\_env\\_default](#), [dataquieR.study\\_data\\_cache\\_quick\\_fill](#)

---

[dataquieR.print\\_block\\_load\\_factor](#)  
*numeric*

---

## Description

multiply size of parallel compute blocks by this factor. the higher it is set, the less smooth progress bar grows, but setting it to a huge number can really speed up the rendering process by approx. 10%. Either set to 1 for full progress control or large (e.g., 1000000) for maximum speed.

## See Also

Other options: [dataquieR](#), [dataquieR.CONDITIONS\\_LEVEL\\_TRHESHOLD](#), [dataquieR.CONDITIONS\\_WITH\\_STACKTRACE](#),  
[dataquieR.ELEMENT\\_MISSMATCH\\_CHECKTYPE](#), [dataquieR.ERRORS\\_WITH\\_CALLER](#), [dataquieR.GAM\\_for\\_LOESS](#),  
[dataquieR.MAHALANOBIS\\_THRESHOLD](#), [dataquieR.MAX\\_LABEL\\_LEN](#), [dataquieR.MAX\\_LONG\\_LABEL\\_LEN](#),  
[dataquieR.MAX\\_VALUE\\_LABEL\\_LEN](#), [dataquieR.MESSAGES\\_WITH\\_CALLER](#), [dataquieR.MULTIVARIATE\\_OUTLIER\\_CHECK](#),  
[dataquieR.VALUE\\_LABELS\\_htmlescaped](#), [dataquieR.WARNINGS\\_WITH\\_CALLER](#), [dataquieR.acc\\_loess.exclude\\_const](#),  
[dataquieR.acc\\_loess.mark\\_time\\_points](#), [dataquieR.acc\\_loess.min\\_bw](#), [dataquieR.acc\\_loess.min\\_proportion](#),  
[dataquieR.acc\\_loess.plot\\_format](#), [dataquieR.acc\\_loess.plot\\_observations](#), [dataquieR.acc\\_margins\\_num](#),  
[dataquieR.acc\\_margins\\_sort](#), [dataquieR.acc\\_multivariate\\_outlier.scale](#), [dataquieR.col\\_con\\_con\\_empirical](#),  
[dataquieR.col\\_con\\_con\\_logical](#), [dataquieR.convert\\_to\\_list\\_for\\_lapply](#), [dataquieR.debug](#),  
[dataquieR.des\\_summary\\_hard\\_lim\\_remove](#), [dataquieR.dontwrapresults](#), [dataquieR.droplevels\\_ReportSummaryT](#),  
[dataquieR.dt\\_adjust](#), [dataquieR.fix\\_column\\_type\\_on\\_read](#), [dataquieR.flip\\_mode](#), [dataquieR.force\\_item\\_spec](#),  
[dataquieR.force\\_label\\_col](#), [dataquieR.grading\\_formats](#), [dataquieR.grading\\_rulesets](#),  
[dataquieR.guess\\_character](#), [dataquieR.guess\\_missing\\_codes](#), [dataquieR.ignore\\_empty\\_vars](#),  
[dataquieR.lang](#), [dataquieR.lazy\\_plots](#), [dataquieR.lazy\\_plots\\_cache](#), [dataquieR.lazy\\_plots\\_gg\\_compatibility](#),  
[dataquieR.locale](#), [dataquieR.max\\_cat\\_resp\\_var\\_levels\\_in\\_plot](#), [dataquieR.max\\_group\\_var\\_levels\\_in\\_plot](#),  
[dataquieR.max\\_group\\_var\\_levels\\_with\\_violins](#), [dataquieR.min\\_obs\\_per\\_group\\_var\\_in\\_plot](#),  
[dataquieR.min\\_time\\_points\\_for\\_cat\\_resp\\_var](#), [dataquieR.non\\_disclosure](#), [dataquieR.old\\_factor\\_handling](#),  
[dataquieR.old\\_type\\_adjust](#), [dataquieR.precomputeStudyData](#), [dataquieR.progress\\_fkt\\_default](#),  
[dataquieR.progress\\_msg\\_fkt\\_default](#), [dataquieR.resume\\_checkpoint](#), [dataquieR.resume\\_print](#),  
[dataquieR.scale\\_level\\_heuristics\\_control\\_binaryrecodeLimit](#), [dataquieR.scale\\_level\\_heuristics\\_control](#),  
[dataquieR.study\\_data\\_cache\\_max](#), [dataquieR.study\\_data\\_cache\\_metrics](#), [dataquieR.study\\_data\\_cache\\_metrics](#),  
[dataquieR.study\\_data\\_cache\\_quick\\_fill](#), [dataquieR.study\\_data\\_colnames\\_case\\_sensitive](#),  
[dataquieR.testdebug](#), [dataquieR.traceback](#), [dataquieR.type\\_adjust\\_parallel](#), [progress\\_init\\_fkt](#)  
 Other `study_data_cache`: [dataquieR.ignore\\_empty\\_vars](#), [dataquieR.precomputeStudyData](#),  
[dataquieR.study\\_data\\_cache\\_max](#), [dataquieR.study\\_data\\_cache\\_metrics](#), [dataquieR.study\\_data\\_cache\\_metrics](#),  
[dataquieR.study\\_data\\_cache\\_metrics\\_env\\_default](#), [dataquieR.study\\_data\\_cache\\_quick\\_fill](#)

---

dataquieR.progress\_fkt\_default

*function to call on progress increase*

---

### Description

TODO

### See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodeLimit`, `dataquieR.scale_level_heuristics_control`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metric`, `dataquieR.study_data_cache_quick_fill`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.progress\_msg\_fkt\_default

*function to call on progress message update*

---

### Description

TODO

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodeLimit`, `dataquieR.scale_level_heuristics_control`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metric`, `dataquieR.study_data_cache_quick_fill`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `dataquieR.progress_init_fkt`

---

`dataquieR.resume_checkpoint`

*If result already exists in a storr back-end, re-use it*

---

**Description**

if TRUE, computation won't be repeated, if a result already exist in the output storr

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`

dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metric, dataquieR.study\_data\_cache\_quick\_fill, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.resume\_print

*If output folder is not empty, try to resume stopped print()*

---

## Description

if TRUE and a report was already partially printed with also this option TRUE, then, a second call to print() will resume the printing.

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISSMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metric, dataquieR.study\_data\_cache\_quick\_fill, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.scale\_level\_heuristics\_control\_binaryrecoodelimit

*Number of levels to consider a variable ordinal in absence of  
SCALE\_LEVEL*

---

### Description

If SCALE\_LEVEL is not specified in the meta\_data, it will be inferred using a heuristic. This option defines, for numeric variables, the maximum number of distinct data values for a variable to be considered ordinal.

### See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_THRESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_c, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.scale\_level\_heuristics\_control\_metriclevels

*Number of levels to consider a variable metric in absence of  
SCALE\_LEVEL*

---

### Description

If SCALE\_LEVEL is not specified in the meta\_data, it will be inferred using a heuristic. This option defines, for numeric variables, the maximum number of distinct data values for a variable to be considered categorical, not ordinal.

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecode`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metric`, `dataquieR.study_data_cache_quick_fill`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `dataquieR.type_adjust_parallel`, `dataquieR.progress_init_fkt`

---

`dataquieR.study_data_cache_max`

*Maximum size of cache for curated study data*

---

**Description**

`dataquieR` caches all used flavors of curated study data, e.g., having missing codes replaced by NAs, having hard limits replaced by NA, ... For larger sets of study data this can be very RAM consuming, so you can control here the maximum size for this cache. Also, this cache is distributed to all compute nodes in case of parallel computation, which may be very time-consuming, and, on single-node-parallelization, also, it may be even more RAM-consuming then.

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`,

dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_quick\_fill, dataquieR.study\_data\_c, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

Other study\_data\_cache: dataquieR.ignore\_empty\_vars, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_met, dataquieR.study\_data\_cache\_metrics\_env\_default, dataquieR.study\_data\_cache\_quick\_fill

---

dataquieR.study\_data\_cache\_metrics

*Collect metrics on cache usage of study data cache*

---

## Description

if TRUE, collect metrics on the usage of the study data cache described here: [dataquieR.study\\_data\\_cache\\_max](#). Won't work, fully, if running in parallel. **[Experimental]**

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot,

dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_quick\_fill, dataquieR.study\_data\_colnames, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

Other study\_data\_cache: dataquieR.ignore\_empty\_vars, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_metrics\_env\_default, dataquieR.study\_data\_cache\_quick\_fill

---

dataquieR.study\_data\_cache\_metrics\_env

*environment for storing metrics on the study data cache*

---

## Description

this is the environment, where metrics will be stored, if `dataquieR.study_data_cache_metrics-option()` has been set TRUE. **[Experimental]**

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_constant, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specification, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_quick\_fill, dataquieR.study\_data\_colnames, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

Other study\_data\_cache: [dataquieR.ignore\\_empty\\_vars](#), [dataquieR.precomputeStudyData](#), [dataquieR.print\\_block\\_load\\_factor](#), [dataquieR.study\\_data\\_cache\\_max](#), [dataquieR.study\\_data\\_cache\\_metrics](#), [dataquieR.study\\_data\\_cache\\_metrics\\_env\\_default](#), [dataquieR.study\\_data\\_cache\\_quick\\_fill](#)

---

`dataquieR.study_data_cache_metrics_env_default`

*Default space for some metrics during report computation*

---

## Description

**[Experimental]**

## Usage

`dataquieR.study_data_cache_metrics_env_default`

## Format

An object of class environment of length 0.

## See Also

Other study\_data\_cache: [dataquieR.ignore\\_empty\\_vars](#), [dataquieR.precomputeStudyData](#), [dataquieR.print\\_block\\_load\\_factor](#), [dataquieR.study\\_data\\_cache\\_max](#), [dataquieR.study\\_data\\_cache\\_metrics](#), [dataquieR.study\\_data\\_cache\\_metrics\\_env](#), [dataquieR.study\\_data\\_cache\\_quick\\_fill](#)

---

`dataquieR.study_data_cache_quick_fill`

*Control the pre-computation of curation levels of study data*

---

## Description

as described in [dataquieR.precomputeStudyData](#), different flavors of the study data are cached. With this option, you control, if before a report is computed, only frequently needed bunch of such flavors are pre-computed, or simply all possible flavors. Won't have any effect, if pre-computation has been turned off.

## See Also

Other options: [dataquieR](#), [dataquieR.CONDITIONS\\_LEVEL\\_TRHESHOLD](#), [dataquieR.CONDITIONS\\_WITH\\_STACKTRACE](#), [dataquieR.ELEMENT\\_MISMATCH\\_CHECKTYPE](#), [dataquieR.ERRORS\\_WITH\\_CALLER](#), [dataquieR.GAM\\_for\\_LOESS](#), [dataquieR.MAHALANOBIS\\_THRESHOLD](#), [dataquieR.MAX\\_LABEL\\_LEN](#), [dataquieR.MAX\\_LONG\\_LABEL\\_LEN](#), [dataquieR.MAX\\_VALUE\\_LABEL\\_LEN](#), [dataquieR.MESSAGES\\_WITH\\_CALLER](#), [dataquieR.MULTIVARIATE\\_OUTLIER\\_CHECK](#), [dataquieR.VALUE\\_LABELS\\_htmlescaped](#), [dataquieR.WARNINGS\\_WITH\\_CALLER](#), [dataquieR.acc\\_loess.exclude\\_const](#), [dataquieR.acc\\_loess.mark\\_time\\_points](#), [dataquieR.acc\\_loess.min\\_bw](#), [dataquieR.acc\\_loess.min\\_proportion](#), [dataquieR.acc\\_loess.plot\\_format](#), [dataquieR.acc\\_loess.plot\\_observations](#), [dataquieR.acc\\_margins\\_num](#),

dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

Other study\_data\_cache: dataquieR.ignore\_empty\_vars, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache\_metrics\_env\_default

---

dataquieR.study\_data\_colnames\_case\_sensitive

*character* Are column names in study data considered case-sensitive for mapping

---

## Description

if TRUE, colnames(study\_data) replaced by the capitalization used in the metadata using a case-insensitive matching, first.

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot,

dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.testdebug, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.testdebug	<i>Disable all interactively used metadata-based function argument provision</i>
---------------------	--

---

## Description

TODO

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE, dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS, dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN, dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.WARNINGS\_WITH\_CALLER, dataquieR.acc\_loess.exclude\_const, dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion, dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num, dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical, dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT, dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec, dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets, dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars, dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility, dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot, dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling, dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor, dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint, dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max, dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache, dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.traceback, dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.traceback     *Include full trace-back in captured conditions*

---

### Description

Caveat: Needs really much memory

### See Also

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlscaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_const`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodelimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

dataquieR.type\_adjust\_parallel

*character* try to do type adjustments in parallel only, if `dq_report2()` was called with `cores = 2` or higher.

---

### Description

`character` try to do type adjustments in parallel only, if `dq_report2()` was called with `cores = 2` or higher.

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.VALUE_LABELS_htmlescaped`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_constant_subgroups`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`, `dataquieR.convert_to_list_for_lapply`, `dataquieR.debug`, `dataquieR.des_summary_hard_lim_remove`, `dataquieR.dontwrapresults`, `dataquieR.droplevels_ReportSummaryT`, `dataquieR.dt_adjust`, `dataquieR.fix_column_type_on_read`, `dataquieR.flip_mode`, `dataquieR.force_item_spec`, `dataquieR.force_label_col`, `dataquieR.grading_formats`, `dataquieR.grading_rulesets`, `dataquieR.guess_character`, `dataquieR.guess_missing_codes`, `dataquieR.ignore_empty_vars`, `dataquieR.lang`, `dataquieR.lazy_plots`, `dataquieR.lazy_plots_cache`, `dataquieR.lazy_plots_gg_compatibility`, `dataquieR.locale`, `dataquieR.max_cat_resp_var_levels_in_plot`, `dataquieR.max_group_var_levels_in_plot`, `dataquieR.max_group_var_levels_with_violins`, `dataquieR.min_obs_per_group_var_in_plot`, `dataquieR.min_time_points_for_cat_resp_var`, `dataquieR.non_disclosure`, `dataquieR.old_factor_handling`, `dataquieR.old_type_adjust`, `dataquieR.precomputeStudyData`, `dataquieR.print_block_load_factor`, `dataquieR.progress_fkt_default`, `dataquieR.progress_msg_fkt_default`, `dataquieR.resume_checkpoint`, `dataquieR.resume_print`, `dataquieR.scale_level_heuristics_control_binaryrecodeLimit`, `dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`, `dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`, `dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`, `progress_init_fkt`

---

`dataquieR.VALUE_LABELS_htmlescaped`

*Assume, all VALUE\_LABELS are  
 Rhref<https://www.w3.org/International/questions/qa-escapesHTML>  
 escaped*

---

**Description**

TODO

**See Also**

Other options: `dataquieR`, `dataquieR.CONDITIONS_LEVEL_TRHESHOLD`, `dataquieR.CONDITIONS_WITH_STACKTRACE`, `dataquieR.ELEMENT_MISSMATCH_CHECKTYPE`, `dataquieR.ERRORS_WITH_CALLER`, `dataquieR.GAM_for_LOESS`, `dataquieR.MAHALANOBIS_THRESHOLD`, `dataquieR.MAX_LABEL_LEN`, `dataquieR.MAX_LONG_LABEL_LEN`, `dataquieR.MAX_VALUE_LABEL_LEN`, `dataquieR.MESSAGES_WITH_CALLER`, `dataquieR.MULTIVARIATE_OUTLIER_CHECK`, `dataquieR.WARNINGS_WITH_CALLER`, `dataquieR.acc_loess.exclude_constant_subgroups`, `dataquieR.acc_loess.mark_time_points`, `dataquieR.acc_loess.min_bw`, `dataquieR.acc_loess.min_proportion`, `dataquieR.acc_loess.plot_format`, `dataquieR.acc_loess.plot_observations`, `dataquieR.acc_margins_num`, `dataquieR.acc_margins_sort`, `dataquieR.acc_multivariate_outlier.scale`, `dataquieR.col_con_con_empirical`, `dataquieR.col_con_con_logical`

dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug, dataquieR.des\_summary\_hard\_lim\_remove,  
 dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryTable, dataquieR.dt\_adjust,  
 dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_specific\_missing\_codes,  
 dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets,  
 dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars,  
 dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility,  
 dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot,  
 dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot,  
 dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling,  
 dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor,  
 dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint,  
 dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit,  
 dataquieR.scale\_level\_heuristics\_control\_metriclevels, dataquieR.study\_data\_cache\_max,  
 dataquieR.study\_data\_cache\_metrics, dataquieR.study\_data\_cache\_metrics\_env, dataquieR.study\_data\_cache,  
 dataquieR.study\_data\_colnames\_case\_sensitive, dataquieR.testdebug, dataquieR.traceback,  
 dataquieR.type\_adjust\_parallel, progress\_init\_fkt

---

dataquieR.WARNINGS\_WITH\_CALLER

*Set caller for warning conditions (to be deprecated)*

---

## Description

to be deprecated

## See Also

Other options: dataquieR, dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, dataquieR.CONDITIONS\_WITH\_STACKTRACE,  
 dataquieR.ELEMENT\_MISSMATCH\_CHECKTYPE, dataquieR.ERRORS\_WITH\_CALLER, dataquieR.GAM\_for\_LOESS,  
 dataquieR.MAHALANOBIS\_THRESHOLD, dataquieR.MAX\_LABEL\_LEN, dataquieR.MAX\_LONG\_LABEL\_LEN,  
 dataquieR.MAX\_VALUE\_LABEL\_LEN, dataquieR.MESSAGES\_WITH\_CALLER, dataquieR.MULTIVARIATE\_OUTLIER\_CHECK,  
 dataquieR.VALUE\_LABELS\_htmlescaped, dataquieR.acc\_loess.exclude\_constant\_subgroups,  
 dataquieR.acc\_loess.mark\_time\_points, dataquieR.acc\_loess.min\_bw, dataquieR.acc\_loess.min\_proportion,  
 dataquieR.acc\_loess.plot\_format, dataquieR.acc\_loess.plot\_observations, dataquieR.acc\_margins\_num,  
 dataquieR.acc\_margins\_sort, dataquieR.acc\_multivariate\_outlier.scale, dataquieR.col\_con\_con\_empirical,  
 dataquieR.col\_con\_con\_logical, dataquieR.convert\_to\_list\_for\_lapply, dataquieR.debug,  
 dataquieR.des\_summary\_hard\_lim\_remove, dataquieR.dontwrapresults, dataquieR.droplevels\_ReportSummaryT,  
 dataquieR.dt\_adjust, dataquieR.fix\_column\_type\_on\_read, dataquieR.flip\_mode, dataquieR.force\_item\_spec,  
 dataquieR.force\_label\_col, dataquieR.grading\_formats, dataquieR.grading\_rulesets,  
 dataquieR.guess\_character, dataquieR.guess\_missing\_codes, dataquieR.ignore\_empty\_vars,  
 dataquieR.lang, dataquieR.lazy\_plots, dataquieR.lazy\_plots\_cache, dataquieR.lazy\_plots\_gg\_compatibility,  
 dataquieR.locale, dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, dataquieR.max\_group\_var\_levels\_in\_plot,  
 dataquieR.max\_group\_var\_levels\_with\_violins, dataquieR.min\_obs\_per\_group\_var\_in\_plot,  
 dataquieR.min\_time\_points\_for\_cat\_resp\_var, dataquieR.non\_disclosure, dataquieR.old\_factor\_handling,  
 dataquieR.old\_type\_adjust, dataquieR.precomputeStudyData, dataquieR.print\_block\_load\_factor,  
 dataquieR.progress\_fkt\_default, dataquieR.progress\_msg\_fkt\_default, dataquieR.resume\_checkpoint,  
 dataquieR.resume\_print, dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit,

`dataquieR.scale_level_heuristics_control_metriclevels`, `dataquieR.study_data_cache_max`,  
`dataquieR.study_data_cache_metrics`, `dataquieR.study_data_cache_metrics_env`, `dataquieR.study_data_cache`,  
`dataquieR.study_data_colnames_case_sensitive`, `dataquieR.testdebug`, `dataquieR.traceback`,  
`dataquieR.type_adjust_parallel`, `progress_init_fkt`

---

`dataquieR_resultset`    *Internal constructor for the internal class `dataquieR_resultset`.*

---

### Description

creates an object of the class `dataquieR_resultset`.

### Usage

```
dataquieR_resultset(...)
```

### Arguments

...                    properties stored in the object

### Details

The class features the following methods:

- `as.data.frame.dataquieR_resultset`, \* `as.list.dataquieR_resultset`, \* `print.dataquieR_resultset`, \* `summary.dataquieR_resultset`

### Value

an object of the class `dataquieR_resultset`.

### See Also

[dq\\_report](#)

---

`dataquieR_resultset2-class`  
*Class `dataquieR_resultset2`.*

---

### Description

Class `dataquieR_resultset2`.

### See Also

[dq\\_report2](#)

---

dataquieR\_resultset\_verify

*Verify an object of class dataquieR\_resultset*

---

### Description

Deprecated

### Usage

dataquieR\_resultset\_verify(...)

### Arguments

...                      Deprecated

### Value

Deprecated

---

DATA\_PREPARATION

*Cross-item level metadata attribute name*

---

### Description

For contradiction rules, the required pre-processing steps that can be given. Note: MISSING\_LABEL, MISSING\_INTERPRET may not work for non-factor variables

### Usage

DATA\_PREPARATION

### Format

An object of class character of length 1.

### Details

LABEL LIMITS MISSING\_NA MISSING\_LABEL MISSING\_INTERPRET

### See Also

[meta\\_data\\_cross](#)

Other meta\_data\_cross: ASSOCIATION\_DIRECTION, ASSOCIATION\_FORM, ASSOCIATION\_METRIC, ASSOCIATION\_RANGE, CHECK\_ID, CHECK\_LABEL, COMPUTED\_VARIABLE\_ROLES, CONTRADICTION\_TERM, CONTRADICTION\_TYPE, GOLDSTANDARD, IRV, MAHALANOBIS\_THRESHOLD, MAXIMUM\_LONG\_STRING, MISS\_RESP, MULTIVARIATE\_OUTLIER\_CHECK, MULTIVARIATE\_OUTLIER\_CHECKTYPE, RELCOMPL\_SPEED, REL\_VAL, RESPT\_PER\_ITEM, SCALE\_ACRONYM, SCALE\_NAME, TOTRESPT, VARIABLE\_LIST, VARIABLE\_LIST\_ORDER, meta\_data\_computation, meta\_data\_cross

---

DATA\_TYPES

*Data Types*

---

## Description

### Data Types of Study Data:

In the metadata, the following entries are allowed for the [variable attribute DATA\\_TYPE](#):

## Usage

DATA\_TYPES

## Format

An object of class `list` of length 5.

## Details

- `integer` for integer numbers
- `string` for text/string/character data
- `float` for decimal/floating point numbers
- `datetime` for timepoints
- `time` for time of day

### Data Types of Function Arguments:

As function arguments, [dataquieR](#) uses additional type specifications:

- `numeric` is a numerical value ([float](#) or [integer](#)), but it is not an allowed `DATA_TYPE` in the metadata. However, some functions may accept `float` or `integer` for specific function arguments. This is, where we use the term `numeric`.
- `enum` allows one element out of a set of allowed options similar to [match.arg](#)
- `set` allows a subset out of a set of allowed options similar to [match.arg](#) with several `.ok = TRUE`.
- `variable` Function arguments of this type expect a character scalar that specifies one variable using the variable identifier given in the metadata attribute `VAR_NAMES` or, if `label_col` is set, given in the metadata attribute given in that argument. Labels can easily be translated using [prep\\_map\\_labels](#)
- `variable list` Function arguments of this type expect a character vector that specifies variables using the variable identifiers given in the metadata attribute `VAR_NAMES` or, if `label_col` is set, given in the metadata attribute given in that argument. Labels can easily be translated using [prep\\_map\\_labels](#)

## See Also

[integer string](#)

---

DATA\_TYPES\_OF\_R\_TYPE *All available data types, mapped from their respective R types*

---

### Description

All available data types, mapped from their respective R types

### Usage

```
DATA_TYPES_OF_R_TYPE
```

### Format

An object of class list of length 17.

### See Also

[prep\\_dq\\_data\\_type\\_of](#)

---

des\_scatterplot\_matrix  
*Compute Pairwise Correlations*

---

### Description

works on variable groups (cross-item\_level), which are expected to show a Pearson correlation

### Usage

```
des_scatterplot_matrix(
  label_col,
  study_data,
  item_level = "item_level",
  meta_data_cross_item = "cross-item_level",
  meta_data = item_level,
  meta_data_v2,
  cross_item_level,
  `cross-item_level`
)
```

**Arguments**

label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data_cross_item	<a href="#">meta_data_cross</a>
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
cross_item_level	<a href="#">data.frame</a> alias for meta_data_cross_item
'cross-item_level'	<a href="#">data.frame</a> alias for meta_data_cross_item

**Details**

[Descriptor](#) # TODO: This can be an indicator

**Value**

a list with the slots:

- SummaryPlotList: for each variable group a [ggplot2::ggplot](#) object with pairwise correlation plots
- SummaryData: table with columns VARIABLE\_LIST, cors, max\_cor, min\_cor
- SummaryTable: like SummaryData, but machine readable and with stable column names.

**Examples**

```
## Not run:
devtools::load_all()
prep_load_workbook_like_file("meta_data_v2")
des_scatterplot_matrix("study_data")

## End(Not run)
```

---

des\_summary

*Compute Descriptive Statistics*


---

**Description**

generates a descriptive overview of the variables in resp\_vars.

[Descriptor](#)

**Usage**

```

des_summary(
  resp_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2,
  hard_limits_removal = getOption("dataquieR.des_summary_hard_lim_remove",
    dataquieR.des_summary_hard_lim_remove_default),
  ...
)

```

**Arguments**

resp_vars	<a href="#">variable</a> the name of the measurement variables
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
hard_limits_removal	<a href="#">logical</a> if TRUE values outside hard limits are removed from the data before calculating descriptive statistics. The default is FALSE
...	arguments to be passed to all called indicator functions if applicable.

**Details**

TODO

**Value**

a [list](#) with:

- SummaryTable: [data.frame](#)
- SummaryData: [data.frame](#)

**See Also**

[Online Documentation](#)

**Examples**

```
## Not run:
xx <- des_summary(study_data = "study_data", meta_data_v2 = "meta_data_v2")
xx$SummaryData

## End(Not run)
```

---

des\_summary\_categorical

*Compute Descriptive Statistics - categorical variables*

---

**Description**

generates a descriptive overview of the categorical variables (nominal and ordinal) in resp\_vars.

[Descriptor](#)

**Usage**

```
des_summary_categorical(
  resp_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2,
  hard_limits_removal = getOption("dataquieR.des_summary_hard_lim_remove",
    dataquieR.des_summary_hard_lim_remove_default),
  ...
)
```

**Arguments**

resp_vars	<a href="#">variable</a> the name of the categorical measurement variable
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
hard_limits_removal	<a href="#">logical</a> if TRUE values outside hard limits are removed from the data before calculating descriptive statistics. The default is FALSE
...	arguments to be passed to all called indicator functions if applicable.

**Details**

TODO

**Value**

a *list* with:

- SummaryTable: [data.frame](#)
- SummaryData: [data.frame](#)

**See Also**

[Online Documentation](#)

**Examples**

```
## Not run:
prep_load_workbook_like_file("meta_data_v2")
xx <- des_summary_categorical(study_data = "study_data", meta_data =
                             prep_get_data_frame("item_level"))
util_html_table(xx$SummaryData)
util_html_table(des_summary_categorical(study_data = prep_get_data_frame("study_data"),
                                       meta_data = prep_get_data_frame("item_level"))$SummaryData)

## End(Not run)
```

---

des\_summary\_continuous

*Compute Descriptive Statistics - continuous variables*

---

**Description**

generates a descriptive overview of continuous variables (ratio and interval) in resp\_vars.

[Descriptor](#)

**Usage**

```
des_summary_continuous(
  resp_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2,
  hard_limits_removal = getOption("dataquieR.des_summary_hard_lim_remove",
  dataquieR.des_summary_hard_lim_remove_default),
  ...
)
```

**Arguments**

resp_vars	<a href="#">variable</a> the name of the continuous measurement variable
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
hard_limits_removal	<a href="#">logical</a> if TRUE values outside hard limits are removed from the data before calculating descriptive statistics. The default is FALSE
...	arguments to be passed to all called indicator functions if applicable.

**Details**

TODO

**Value**

a [list](#) with:

- SummaryTable: [data.frame](#)
- SummaryData: [data.frame](#)

**See Also**

[Online Documentation](#)

**Examples**

```
## Not run:
prep_load_workbook_like_file("meta_data_v2")
xx <- des_summary_continuous(study_data = "study_data", meta_data =
                             prep_get_data_frame("item_level"))
xx$SummaryData
## End(Not run)
```

---

DF_CODE	<i>Data frame level metadata attribute name</i>
---------	---

---

**Description**

Name of the data frame

**Usage**

DF\_CODE

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_dataframe](#)

---

DF_ELEMENT_COUNT	<i>Data frame level metadata attribute name</i>
------------------	---

---

**Description**

Number of expected data elements in a data frame. [numeric](#). Check only conducted if number entered

**Usage**

DF\_ELEMENT\_COUNT

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_dataframe](#)

---

DF_ID_REF_TABLE	<i>Data frame level metadata attribute name</i>
-----------------	---

---

**Description**

The name of the data frame containing the reference IDs to be compared with the IDs in the study data set.

**Usage**

DF\_ID\_REF\_TABLE

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_dataframe](#)

---

DF_ID_VARS	<i>Data frame level metadata attribute name</i>
------------	---

---

**Description**

All variables that are to be used as one single ID variable (combined key) in a data frame.

**Usage**

DF\_ID\_VARS

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_dataframe](#)

DF\_NAME *Data frame level metadata attribute name*

---

**Description**

Name of the data frame

**Usage**

DF\_NAME

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_dataframe](#)

---

DF\_RECORD\_CHECK *Data frame level metadata attribute name*

---

**Description**

The type of check to be conducted when comparing the reference ID table with the IDs delivered in the study data files.

**Usage**

DF\_RECORD\_CHECK

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_dataframe](#)

---

DF_RECORD_COUNT	<i>Data frame level metadata attribute name</i>
-----------------	---

---

**Description**

Number of expected data records in a data frame. [numeric](#). Check only conducted if number entered

**Usage**

DF\_RECORD\_COUNT

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_dataframe](#)

---

DF_UNIQUE_ID	<i>Data frame level metadata attribute name</i>
--------------	---

---

**Description**

Defines expectancies on the uniqueness of the IDs across the rows of a data frame, or the number of times some ID can be repeated.

**Usage**

DF\_UNIQUE\_ID

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_dataframe](#)

---

DF_UNIQUE_ROWS	<i>Data frame level metadata attribute name</i>
----------------	---

---

**Description**

Specifies whether identical data is permitted across rows in a data frame (excluding ID variables)

**Usage**

```
DF_UNIQUE_ROWS
```

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_dataframe](#)

---

dim.dataquieR_resultset2	<i>Get the dimensions of a dq_report2 result</i>
--------------------------	--

---

**Description**

Get the dimensions of a dq\_report2 result

**Usage**

```
## S3 method for class 'dataquieR_resultset2'  
dim(x)
```

**Arguments**

x                    a dataquieR\_resultset2 result

**Value**

dimensions

---

dimensions	<i>Names of DQ dimensions</i>
------------	-------------------------------

---

**Description**

a vector of data quality dimensions. The supported dimensions are Completeness, Consistency and Accuracy.

**Usage**

```
dimensions
```

**Format**

An object of class character of length 3.

**Value**

Only a definition, not a function, so no return value

**See Also**

[Data Quality Concept](#)

---

dimnames.dataquieR_resultset2	<i>Names of a dataquieR report object (v2.0)</i>
-------------------------------	--

---

**Description**

Names of a dataquieR report object (v2.0)

**Usage**

```
## S3 method for class 'dataquieR_resultset2'
dimnames(x)
```

**Arguments**

x                    the result object

**Value**

the names

---

dims

*Dimension Titles for Prefixes*

---

**Description**

order does matter, because it defines the order in the dq\_report2.

**Usage**

dims

**Format**

An object of class character of length 5.

**See Also**

util\_html\_for\_var()  
util\_html\_for\_dims()

---

DISTRIBUTIONS

*All available probability distributions for [acc\\_shape\\_or\\_scale](#)*

---

**Description**

- uniform For uniform distribution
- normal For Gaussian distribution
- gamma For a gamma distribution

**Usage**

DISTRIBUTIONS

**Format**

An object of class list of length 3.

---

dq_report	<i>Generate a full DQ report</i>
-----------	----------------------------------

---

**Description**

Deprecated

**Usage**

```
dq_report(...)
```

**Arguments**

...           Deprecated

**Value**

Deprecated

---

dq_report2	<i>Generate a full DQ report, v2</i>
------------	--------------------------------------

---

**Description**

Generate a full DQ report, v2

**Usage**

```
dq_report2(
  study_data,
  item_level = "item_level",
  label_col = LABEL,
  meta_data_segment = "segment_level",
  meta_data_dataframe = "dataframe_level",
  meta_data_cross_item = "cross-item_level",
  meta_data_item_computation = "item_computation_level",
  meta_data = item_level,
  meta_data_v2,
  ...,
  dimensions = c("Completeness", "Consistency"),
  cores = list(mode = "socket", logging = FALSE, cpus = util_detect_cores(),
    load.balancing = TRUE),
  ignore_empty_vars = getOption("dataquieR.ignore_empty_vars",
    dataquieR.ignore_empty_vars_default),
  specific_args = list(),
  advanced_options = list(),
```

```

author = prep_get_user_name(),
title = "Data quality report",
subtitle = as.character(Sys.Date()),
user_info = NULL,
debug_parallel = FALSE,
resp_vars = character(0),
filter_indicator_functions = character(0),
filter_result_slots = c("^Summary", "^Segment", "^DataTypePlotList",
  "ReportSummaryTable", "Dataframe", "Result", "VariableGroup"),
mode = c("default", "futures", "queue", "parallel"),
mode_args = list(),
notes_from_wrapper = list(),
storr_factory = NULL,
amend = FALSE,
cross_item_level,
`cross-item_level`,
segment_level,
dataframe_level,
item_computation_level,
.internal = rlang::env_inherits(rlang::caller_env(), parent.env(environment())),
checkpoint_resumed = getOption("dataquieR.resume_checkpoint",
  dataquieR.resume_checkpoint_default),
name_of_study_data,
dt_adjust = as.logical(getOption("dataquieR.dt_adjust", dataquieR.dt_adjust_default))
)

```

### Arguments

**study\_data**      [data.frame](#) the data frame that contains the measurements  
**item\_level**      [data.frame](#) the data frame that contains metadata attributes of study data  
**label\_col**        [variable attribute](#) the name of the column in the metadata with labels of variables  
**meta\_data\_segment**  
                     [data.frame](#) – optional: Segment level metadata  
**meta\_data\_dataframe**  
                     [data.frame](#) – optional: Data frame level metadata  
**meta\_data\_cross\_item**  
                     [data.frame](#) – optional: Cross-item level metadata  
**meta\_data\_item\_computation**  
                     [data.frame](#) optional. computation rules for computed variables.  
**meta\_data**        [data.frame](#) old name for item\_level  
**meta\_data\_v2**    [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#)  
                     for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using  
                     [prep\\_purge\\_data\\_frame\\_cache](#), if you specify meta\_data\_v2.  
**...**             arguments to be passed to all called indicator functions if applicable.  
**dimensions**      [dimensions](#) Vector of dimensions to address in the report. Allowed values in  
                     the vector are Completeness, Consistency, and Accuracy. The generated report

will only cover the listed data quality dimensions. Accuracy is computational expensive, so this dimension is not enabled by default. Completeness should be included, if Consistency is included, and Consistency should be included, if Accuracy is included to avoid misleading detections of e.g. missing codes as outliers, please refer to the data quality concept for more details. Integrity is always included. If dimensions is equal to NULL or "all", all dimensions will be covered.

cores	<b>integer</b> number of cpu cores to use or a named list with arguments for <code>parallelMap::parallelStart</code> or NULL, if parallel has already been started by the caller. Can also be a cluster.
ignore_empty_vars	<b>enum</b> TRUE   FALSE   auto. See <code>dataquieR.ignore_empty_vars</code> .
specific_args	<b>list</b> named list of arguments specifically for one of the called functions, the of the list elements correspond to the indicator functions whose calls should be modified. The elements are lists of arguments.
advanced_options	<b>list</b> options to set during report computation, see <code>options()</code>
author	<b>character</b> author for the report documents.
title	<b>character</b> optional argument to specify the title for the data quality report
subtitle	<b>character</b> optional argument to specify a subtitle for the data quality report
user_info	<b>list</b> additional info stored with the report, e.g., comments, title, ...
debug_parallel	<b>logical</b> print blocks currently evaluated in parallel
resp_vars	<b>variable list</b> the name of the measurement variables for the report. If missing, all variables will be used. Only item level indicator functions are filtered, so far.
filter_indicator_functions	<b>character</b> regular expressions, only if an indicator function's name matches one of these, it'll be used for the report. If of length zero, no filtering is performed.
filter_result_slots	<b>character</b> regular expressions, only if an indicator function's result's name matches one of these, it'll be used for the report. If of length zero, no filtering is performed.
mode	<b>character</b> work mode for parallel execution. default is "default", the values mean: - default: use queue except cores has been set explicitly - futures: use the future package - queue: use a queue as described in the examples from the callr package by Csárdi and Chang and start sub-processes as workers that evaluate the queue. - parallel: use the cluster from cores to evaluate all calls of indicator functions using the classic R parallel back-ends
mode_args	<b>list</b> of arguments for the selected mode. As of writing this manual, only for the mode queue the argument step is supported, which gives the number of function calls that are run by one worker at a time. the default is 15, which gives on most of the tested systems a good balance between synchronization overhead and idling workers.
notes_from_wrapper	<b>list</b> a list containing notes about changed labels by <code>dq_report_by</code> (otherwise NULL)

storr_factory	<b>function</b> NULL, or a function returning a storr object as back-end for the report's results. If used with cores > 1, the storage must be accessible from all cores and capable of concurrent writing according to storr. Hint: dataquieR currently only supports storr::storr_rds(), officially, while other back-ends may nevertheless work, yet, they are not tested.
amend	<b>logical</b> if there is already data in.storr_factory, use it anyways – unsupported, so far!
cross_item_level	<b>data.frame</b> alias for meta_data_cross_item
segment_level	<b>data.frame</b> alias for meta_data_segment
dataframe_level	<b>data.frame</b> alias for meta_data_dataframe
item_computation_level	<b>data.frame</b> alias for meta_data_item_computation
.internal	<b>logical</b> internal use, only.
checkpoint_resumed	<b>logical</b> if using a storr_factory and the back-end there is already filled, and if amend is missing or set to TRUE, compute all missing result and add them to the back-end.
name_of_study_data	<b>character</b> name for study data inside the report, internal use.
dt_adjust	<b>logical</b> whether to trust data types in the study data. if TRUE, data types are checked based on the metadata and later casted to the declared type. if your data source is already typed, this can be turned off to speed up computations. see <a href="#">dataquieR.dt_adjust</a>
'cross-item_level'	<b>data.frame</b> alias for meta_data_cross_item

## Details

See [dq\\_report\\_by](#) for a way to generate stratified or splitted reports easily.

## Value

a [dataquieR\\_resultset2](#) that can be [printed](#) creating a HTML-report.

## See Also

- [as.data.frame.dataquieR\\_resultset](#)
- [as.list.dataquieR\\_resultset](#)
- [print.dataquieR\\_resultset](#)
- [summary.dataquieR\\_resultset](#)
- [dq\\_report\\_by](#)

---

dq_report_by	<i>Generate a stratified full DQ report</i>
--------------	---

---

**Description**

Generate a stratified full DQ report

**Usage**

```

dq_report_by(
  study_data,
  item_level = "item_level",
  meta_data_segment = "segment_level",
  meta_data_dataframe = "dataframe_level",
  meta_data_cross_item = "cross-item_level",
  meta_data_item_computation = "item_computation_level",
  missing_tables = NULL,
  label_col,
  meta_data_v2,
  segment_column = NULL,
  strata_column = NULL,
  strata_select = NULL,
  selection_type = NULL,
  segment_select = NULL,
  segment_exclude = NULL,
  strata_exclude = NULL,
  subgroup = NULL,
  resp_vars = character(),
  id_vars = NULL,
  advanced_options = list(),
  storr_factory = NULL,
  amend = FALSE,
  checkpoint_resumed = getOption("dataquieR.resume_checkpoint",
    dataquieR.resume_checkpoint_default),
  ...,
  output_dir = NULL,
  input_dir = NULL,
  also_print = FALSE,
  disable_plotly = FALSE,
  view = TRUE,
  meta_data = item_level,
  cross_item_level,
  `cross-item_level`,
  segment_level,
  dataframe_level,
  item_computation_level
)

```

## Arguments

study_data	<b>data.frame</b> the data frame that contains the measurements: it can be an R object (e.g., <code>bia</code> ), a data frame (e.g., <code>"C:/Users/data/bia.dta"</code> ), a vector containing data frames files (e.g., <code>c("C:/Users/data/bia.dta", "C:/Users/data/biames.dta")</code> ), or it can be left empty and the data frames are provided in the data frame level metadata. If only the file name without path is provided (e.g., <code>"bia.dta"</code> ), the file name needs the extension and the path must be provided in the argument <code>input_dir</code> . It can also contain only the file name in case of example data from the package <code>dataquieR</code> (e.g., <code>"study_data"</code> or <code>"ship"</code> )
item_level	<b>data.frame</b> the data frame that contains metadata attributes of study data
meta_data_segment	<b>data.frame</b> – optional: Segment level metadata
meta_data_dataframe	<b>data.frame</b> – optional if <code>study_data</code> is present: Data frame level metadata
meta_data_cross_item	<b>data.frame</b> – optional: Cross-item level metadata
meta_data_item_computation	<b>data.frame</b> – optional: Computed items metadata
missing_tables	<b>character</b> the name of the data frame containing the missing codes, it can be a vector if more than one table is provided. Example: <code>c("missing_table1", "missing_table2")</code>
label_col	<b>variable attribute</b> the name of the column in the metadata containing the labels of the variables
meta_data_v2	<b>character</b> path or file name of the workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify <code>meta_data_v2</code>
segment_column	<b>variable attribute</b> name of a metadata attribute usable to split the report in sections of variables, e.g. all blood-pressure related variables. By default, reports are split by <code>STUDY_SEGMENT</code> if available and no <code>segment_column</code> nor <code>strata_column</code> or <code>subgroup</code> are defined. To create an un-split report please write explicitly the argument <code>'segment_column = NULL'</code>
strata_column	<b>variable</b> name of a study variable to stratify the report by, e.g. the study centers. Both labels and <code>VAR_NAMES</code> are accepted. In case of NAs in the selected variable, a separate report containing the NAs subset will be created
strata_select	<b>character</b> if given, the strata of <code>strata_column</code> are limited to the content of this vector. A character vector or a regular expression can be provided (e.g., <code>"^a.*\$"</code> ). This argument can not be used if no <code>strata_column</code> is provided
selection_type	<b>character</b> optional, can only be specified if a <code>strata_select</code> or <code>strata_exclude</code> is specified. If not present the function try to guess what the user typed as <code>strata_select</code> or <code>strata_exclude</code> . There are 3 options: value indicating that the stratum selected is a value and not a <code>value_label</code> . For example <code>"0"</code> ; <code>v_label</code> indicating that the stratum specified is a label. For example <code>"male"</code> . <code>regex</code> indicating that the user specified strata using a regular expression. For example <code>"^Ber"</code> to select all strata starting with that letters

segment_select	<b>character</b> if given, the levels of segment_column are limited to the content of this vector. A character vector or a regular expression (e.g., ".*_EXAM\$") can be provided. This argument can not be used if no segment_column is provided.
segment_exclude	<b>character</b> optional, can only be specified if a segment_column is specified. The levels of segment_column will not include the content of this argument. A character vector or a regular expression can be provided (e.g., "^STU").
strata_exclude	<b>character</b> optional, can only be specified if a strata_column is specified. The strata of strata_column will not include the content of this argument. A character vector or a regular expression can be provided (e.g., "^STU").
subgroup	<b>character</b> optional, to define subgroups of cases. Rules are to be written as REDCap rules. Only VAR_NAMES are accepted in the rules.
resp_vars	<b>variable</b> the names of the measurement variables, if missing or NULL, all variables will be included
id_vars	<b>variable</b> a vector containing the name/s of the variables containing ids, to be used to merge multiple data frames if provided in study_data and to be add to referred vars
advanced_options	<b>list</b> options to set during report computation, see <a href="#">options()</a>
storr_factory	<b>function</b> NULL, or a function returning a storr object as back-end for the report's results. If used with cores > 1, the storage must be accessible from all cores and capable of concurrent writing according to storr. Hint: dataquieR currently only supports storr::storr_rds(), officially, while other back-ends may nevertheless work, yet, they are not tested.
amend	<b>logical</b> if there is already data in.storr_factory, use it anyways – unsupported, so far!
checkpoint_resumed	<b>logical</b> if using a storr_factory and the back-end there is already filled, and if amend is missing or set to TRUE, compute all missing result and add them to the back-end.
...	arguments to be passed through to <a href="#">dq_report</a> or <a href="#">dq_report2</a>
output_dir	<b>character</b> if given, the output is not returned but saved in this directory
input_dir	<b>character</b> if given, the study data files that have no path and that are not URL are searched in this directory. Also meta_data_v2 is searched in this directory if no path is provided
also_print	<b>logical</b> if output_dir is not NULL, also create HTML output for each report using <a href="#">print.dataquieR_resultset2()</a> written to the path output_dir
disable_plotly	<b>logical</b> do not use plotly, even if installed
view	<b>logical</b> open the returned report
meta_data	<b>data.frame</b> old name for item_level
cross_item_level	<b>data.frame</b> alias for meta_data_cross_item
segment_level	<b>data.frame</b> alias for meta_data_segment

```

dataframe_level
    data.frame alias for meta_data_dataframe
item_computation_level
    data.frame alias for meta_data_item_computation
'cross-item_level'
    data.frame alias for meta_data_cross_item

```

## Value

A named [list](#) of named [lists](#) of [dq\\_report2](#) reports, returned invisibly unless `view = TRUE`. If `output_dir` is given, the result is still returned (invisibly), and optionally opened in a browser (`view = TRUE`, `also_print = TRUE`).

## See Also

[dq\\_report](#)

## Examples

```

## Not run: # really long-running example.
prep_load_workbook_like_file("meta_data_v2")
rep <- dq_report_by("study_data", label_col =
  LABEL, strata_column = "CENTER_0")
rep <- dq_report_by("study_data",
  label_col = LABEL, strata_column = "CENTER_0",
  segment_column = NULL
)
unlink("/tmp/testRep/", force = TRUE, recursive = TRUE)
dq_report_by("study_data",
  label_col = LABEL, strata_column = "CENTER_0",
  segment_column = STUDY_SEGMENT, output_dir = "/tmp/testRep"
)
unlink("/tmp/testRep/", force = TRUE, recursive = TRUE)
dq_report_by("study_data",
  label_col = LABEL, strata_column = "CENTER_0",
  segment_column = NULL, output_dir = "/tmp/testRep"
)
dq_report_by("study_data",
  label_col = LABEL,
  segment_column = STUDY_SEGMENT, output_dir = "/tmp/testRep"
)
dq_report_by("study_data",
  label_col = LABEL,
  segment_column = STUDY_SEGMENT, output_dir = "/tmp/testRep",
  also_print = TRUE
)
dq_report_by(study_data = "study_data", meta_data_v2 = "meta_data_v2",
  advanced_options = list(dataquieR.study_data_cache_max = 0,
    dataquieR.study_data_cache_metrics = TRUE,
    dataquieR.study_data_cache_metrics_env = environment(),
    cores = NULL, dimensions = "int")
dq_report_by(study_data = "study_data", meta_data_v2 = "meta_data_v2",

```

```
advanced_options = list(dataquieR.study_data_cache_max = 0),
cores = NULL, dimensions = "int")

## End(Not run)
```

---

droplevels.ReportSummaryTable  
*Remove unused levels from ReportSummaryTable*

---

### Description

Remove unused levels from ReportSummaryTable

### Usage

```
## S3 method for class 'ReportSummaryTable'
droplevels(x, ...)
```

### Arguments

x	ReportSummaryTable an object from which to drop unused factor levels.
...	no used.

### Value

ReportSummaryTable with all (NA or 0)-columns removed

---

ggplotGrob.dq\_lazy\_ggplot  
*S3/S7 methods for lazy ggplot objects*

---

### Description

These S3/S7 methods make dq\_lazy\_ggplot/dq\_lazy\_ggplot\_s7 objects work smoothly with functions from **ggplot2** and **plotly**. They simply materialize the underlying ggplot object and then delegate to the respective generic.

**Usage**

```
ggplotGrob.dq_lazy_ggplot(x, ...)
ggplotly.dq_lazy_ggplot_s7(p, ...)
plotly_build.dq_lazy_ggplot_s7(p, ...)
ggplotly.dq_lazy_ggplot(p, ...)
plotly_build.dq_lazy_ggplot(p, ...)
ggplotGrob.dq_lazy_ggplot_s7(x, ...)
```

**Arguments**

<code>x, p</code>	A <code>dq_lazy_ggplot</code> object.
<code>...</code>	Further arguments passed on to the underlying generic.

**Value**

The return value is the same as for the corresponding generic:

- `ggplotGrob()` returns a `gtable` object.
- `ggplotly()` returns a `plotly` object.
- `plotly_build()` returns a `plotly_proxy` or similar.

**See Also**

[ggplotGrob](#), `plotly::ggplotly` `plotly::plotly_build`

---

GOLDSTANDARD

*Cross-item level metadata attribute name*

---

**Description**

Defines the measurement variable to be used as a known gold standard. Only one variable can be defined as the gold standard.

**Usage**

```
GOLDSTANDARD
```

**Format**

An object of class character of length 1.

**See Also**[meta\\_data\\_cross](#)

Other `meta_data_cross`: `ASSOCIATION_DIRECTION`, `ASSOCIATION_FORM`, `ASSOCIATION_METRIC`, `ASSOCIATION_RANGE`, `CHECK_ID`, `CHECK_LABEL`, `COMPUTED_VARIABLE_ROLES`, `CONTRADICTION_TERM`, `CONTRADICTION_TYPE`, `DATA_PREPARATION`, `IRV`, `MAHALANOBIS_THRESHOLD`, `MAXIMUM_LONG_STRING`, `MISS_RESP`, `MULTIVARIATE_OUTLIER_CHECK`, `MULTIVARIATE_OUTLIER_CHECKTYPE`, `RELCOMPL_SPEED`, `REL_VAL`, `RESPT_PER_ITEM`, `SCALE_ACRONYM`, `SCALE_NAME`, `TOTRESPT`, `VARIABLE_LIST`, `VARIABLE_LIST_ORDER`, `meta_data_computation`, `meta_data_cross`

---

```
grid.draw.util_pairs_ggplot_panels
```

```
grid.draw method for util_pairs_ggplot_panels objects
```

---

**Description**

grid.draw method for `util_pairs_ggplot_panels` objects

**Usage**

```
## S3 method for class 'util_pairs_ggplot_panels'
grid.draw(x, ...)
```

**Arguments**

<code>x</code>	An object of class <code>util_pairs_ggplot_panels</code> .
<code>...</code>	Ignored.

---

```
html_dependency_clipboard
```

```
HTML Dependency for report headers in clipboard
```

---

**Description**

HTML Dependency for report headers in clipboard

**Usage**

```
html_dependency_clipboard()
```

**Value**

the dependency

html\_dependency\_dataquieR

*HTML Dependency for dataquieR*

---

### Description

generate all dependencies used in static dataquieR reports

### Usage

```
html_dependency_dataquieR(iframe = FALSE)
```

### Arguments

iframe [logical](#)(1) if TRUE, create the dependency used in figure iframes.

### Value

the dependency

---

html\_dependency\_jspdf *HTML dependency for jsPDF*

---

### Description

Provides jsPDF for use in Shiny or RMarkdown via htmltools.

### Usage

```
html_dependency_jspdf()
```

### Value

An `htmltools::htmlDependency()` object

---

html\_dependency\_report\_dt  
*HTML Dependency for report headers in DT::datatable*

---

**Description**

HTML Dependency for report headers in DT::datatable

**Usage**

html\_dependency\_report\_dt()

**Value**

the dependency

---

html\_dependency\_tippy *HTML Dependency for tippy*

---

**Description**

HTML Dependency for tippy

**Usage**

html\_dependency\_tippy()

**Value**

the dependency

---

html\_dependency\_vert\_dt  
*HTML Dependency for vertical headers in DT::datatable*

---

**Description**

HTML Dependency for vertical headers in DT::datatable

**Usage**

html\_dependency\_vert\_dt()

**Value**

the dependency

---

 int\_all\_datastructure\_dataframe

*Wrapper function to check for studies data structure*


---

## Description

This function tests for unexpected elements and records, as well as duplicated identifiers and content. The unexpected element record check can be conducted by providing the number of expected records or an additional table with the expected records. It is possible to conduct the checks by study segments or to consider only selected segments.

[Indicator](#)

## Usage

```
int_all_datastructure_dataframe(
  meta_data_dataframe = "dataframe_level",
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2,
  dataframe_level
)
```

## Arguments

meta_data_dataframe	<a href="#">data.frame</a> the data frame that contains the metadata for the data frame level
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
dataframe_level	<a href="#">data.frame</a> alias for meta_data_dataframe

## Value

a [list](#) with

- [DataframeTable](#): data frame with selected check results, used for the data quality report.

## Examples

```
## Not run:
out_dataframe <- int_all_datastructure_dataframe(
  meta_data_dataframe = "meta_data_dataframe",
  meta_data = "ship_meta"
)
```

```

md0 <- prep_get_data_frame("ship_meta")
md0
md0$VAR_NAMES
md0$VAR_NAMES[[1]] <- "Id" # is this mismatch reported -- is the data frame
                          # also reported, if nothing is wrong with it
out_dataframe <- int_all_datastructure_dataframe(
  meta_data_dataframe = "meta_data_dataframe",
  meta_data = md0
)

# This is the "normal" procedure for inside pipeline
# but outside this function checktype is exact by default
options(dataquieR.ELEMENT_MISMATCH_CHECKTYPE = "subset_u")
lapply(setNames(nm = prep_get_data_frame("meta_data_dataframe")$DF_NAME),
  int_sts_element_dataframe, meta_data = md0)
md0$VAR_NAMES[[1]] <-
  "id" # is this mismatch reported -- is the data frame also reported,
      # if nothing is wrong with it
lapply(setNames(nm = prep_get_data_frame("meta_data_dataframe")$DF_NAME),
  int_sts_element_dataframe, meta_data = md0)
options(dataquieR.ELEMENT_MISMATCH_CHECKTYPE = "exact")

## End(Not run)

```

---

```
int_all_datastructure_segment
```

*Wrapper function to check for segment data structure*

---

## Description

This function tests for unexpected elements and records, as well as duplicated identifiers and content. The unexpected element record check can be conducted by providing the number of expected records or an additional table with the expected records. It is possible to conduct the checks by study segments or to consider only selected segments.

[Indicator](#)

## Usage

```

int_all_datastructure_segment(
  study_data,
  label_col,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2,
  segment_level,
  meta_data_segment = "segment_level"
)

```

**Arguments**

study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
segment_level	<a href="#">data.frame</a> alias for meta_data_segment
meta_data_segment	<a href="#">data.frame</a> the data frame that contains the metadata for the segment level, mandatory

**Value**

a [list](#) with

- SegmentTable: data frame with selected check results, used for the data quality report.

**Examples**

```
## Not run:
out_segment <- int_all_datastructure_segment(
  meta_data_segment = "meta_data_segment",
  study_data = "ship",
  meta_data = "ship_meta"
)

study_data <- cars
meta_data <- dataquieR::prep_create_meta(VAR_NAMES = c("speedx", "distx"),
  DATA_TYPE = c("integer", "integer"), MISSING_LIST = "|", JUMP_LIST = "|",
  STUDY_SEGMENT = c("Intro", "Ex"))

out_segment <- int_all_datastructure_segment(
  meta_data_segment = "meta_data_segment",
  study_data = study_data,
  meta_data = meta_data
)

## End(Not run)
```

---

int\_datatype\_matrix    *Check declared data types of metadata in study data*

---

**Description**

Checks data types of the study data and for the data type declared in the metadata

[Indicator](#)

**Usage**

```
int_datatype_matrix(
  resp_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  split_segments = FALSE,
  max_vars_per_plot = 20,
  threshold_value = 0,
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

`resp_vars` [variable](#) the names of the measurement variables, if missing or NULL, all variables will be checked

`study_data` [data.frame](#) the data frame that contains the measurements

`label_col` [variable attribute](#) the name of the column in the metadata with labels of variables

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`split_segments` [logical](#) return one matrix per study segment

`max_vars_per_plot` [integer](#) from=0. The maximum number of variables per single plot.

`threshold_value` [numeric](#) from=0 to=100. percentage failing conversions allowed to still classify a study variable convertible.

`meta_data` [data.frame](#) old name for `item_level`

`meta_data_v2` [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

**Details**

This is a preparatory support function that compares study data with associated metadata. A prerequisite of this function is that the no. of columns in the study data complies with the no. of rows in the metadata.

For each study variable, the function searches for its data type declared in static metadata and returns a heatmap like matrix indicating data type mismatches in the study data.

List function.

**Value**

a list with:

- `SummaryTable`: data frame containing data quality check for "data type mismatch" (`CLS_int_vfe_type`, `PCT_int_vfe_type`). The following categories are possible: categories: "Non-matching datatype", "Non-Matching datatype, convertible", "Matching datatype"

- SummaryData: data frame containing data quality check for "data type mismatch" for a report
- SummaryPlot: [ggplot2::ggplot2](#) heatmap plot, graphical representation of SummaryTable
- DataTypePlotList: [list](#) of plots per (maybe artificial) segment
- ReportSummaryTable: data frame underlying SummaryPlot

---

int\_duplicate\_content *Check for duplicated content*

---

### Description

This function tests for duplicated entries in the data set. It is possible to check duplicated entries by study segments or to consider only selected segments.

#### Indicator

### Usage

```
int_duplicate_content(
  level = c("dataframe", "segment"),
  study_data,
  item_level = "item_level",
  label_col,
  meta_data = item_level,
  meta_data_v2,
  ...
)
```

### Arguments

level	<a href="#">character</a> a character vector indicating whether the assessment should be conducted at the study level (level = "dataframe") or at the segment level (level = "segment").
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
...	Depending on level, passed to either <code>util_int_duplicate_content_segment</code> or <code>util_int_duplicate_content_dataframe</code>

### Value

a [list](#). Depending on level, see `util_int_duplicate_content_segment` or `util_int_duplicate_content_dataframe` for a description of the outputs.

---

int\_duplicate\_ids      *Check for duplicated IDs*

---

### Description

This function tests for duplicates entries in identifiers. It is possible to check duplicated identifiers by study segments or to consider only selected segments.

#### Indicator

### Usage

```
int_duplicate_ids(
  level = c("dataframe", "segment"),
  study_data,
  item_level = "item_level",
  label_col,
  meta_data = item_level,
  meta_data_v2,
  ...
)
```

### Arguments

level	<b>character</b> a character vector indicating whether the assessment should be conducted at the study level (level = "dataframe") or at the segment level (level = "segment").
study_data	<b>data.frame</b> the data frame that contains the measurements
item_level	<b>data.frame</b> the data frame that contains metadata attributes of study data
label_col	<b>variable attribute</b> the name of the column in the metadata with labels of variables
meta_data	<b>data.frame</b> old name for item_level
meta_data_v2	<b>character</b> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
...	Depending on level, passed to either <code>util_int_duplicate_ids_segment</code> or <code>util_int_duplicate_ids_dataframe</code>

### Value

a **list**. Depending on level, see `util_int_duplicate_ids_segment` or `util_int_duplicate_ids_dataframe` for a description of the outputs.

---

int\_encoding\_errors    *Encoding Errors*


---

### Description

Detects errors in the character encoding of string variables

[Indicator](#)

### Usage

```
int_encoding_errors(
  resp_vars = NULL,
  study_data,
  label_col,
  meta_data_dataframe = "dataframe_level",
  item_level = "item_level",
  ref_encs,
  meta_data = item_level,
  meta_data_v2,
  dataframe_level
)
```

### Arguments

resp_vars	<a href="#">variable</a> the names of the measurement variables, if missing or NULL, all variables will be checked
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
meta_data_dataframe	<a href="#">data.frame</a> the data frame that contains the metadata for the data frame level
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
ref_encs	reference encodings (names are resp_vars)
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
dataframe_level	<a href="#">data.frame</a> alias for meta_data_dataframe

### Details

Strings are stored based on [code tables](#), nowadays, typically as [UTF-8](#). However, other code systems are still in use, so, sometimes, strings from different systems are mixed in the data. This indicator checks for such problems and returns the count of entries per variable, that do not match

the reference coding system, which is estimated from the study data (addition of metadata field is planned).

If not specified in the metadata (columns ENCODING in item- or data-frame- level, the encoding is guessed from the data). Otherwise, it may be any supported encoding as returned by `iconvlist()`.

### Value

a list with:

- SummaryTable: [data.frame](#) with information on such problems
- SummaryData: [data.frame](#) human readable version of SummaryTable
- FlaggedStudyData: [data.frame](#) tells for each entry in study data if its encoding is OK. has the same dimensions as `study_data`

---

int\_part\_vars\_structure

*Detect Expected Observations*

---

### Description

For each participant, check, if an observation was expected, given the PART\_VARS from item-level metadata

### Usage

```
int_part_vars_structure(
  label_col,
  study_data,
  item_level = "item_level",
  expected_observations = c("HIERARCHY", "SEGMENT"),
  disclose_problem_paprt_var_data = FALSE,
  meta_data = item_level,
  meta_data_v2
)
```

### Arguments

`label_col` [character](#) mapping attribute `colnames(study_data)` vs. `meta_data[label_col]`

`study_data` [study\\_data](#) must have all relevant PART\_VARS to avoid false-positives on PART\_VARS missing from `study_data`

`item_level` [meta\\_data](#) must be complete to avoid false positives on non-existing PART\_VARS

`expected_observations` [enum](#) HIERARCHY | SEGMENT. How should PART\_VARS be handled: - SEGMENT: if PART\_VAR is 1, an observation is expected - HIERARCHY: the default, if the PART\_VAR is 1 for this variable and also for all PART\_VARS of PART\_VARS up in the hierarchy, an observation is expected.

disclose\_problem\_paprt\_var\_data      [logical](#) show the problematic data (PART\_VAR only)

meta\_data      [data.frame](#) old name for item\_level

meta\_data\_v2      [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify meta\_data\_v2.

## Details

[Descriptor](#)

## Value

empty list, so far – the function only warns.

---

int\_sts\_element\_dataframe

*Determine missing and/or superfluous data elements*

---

## Description

Depends on [dataquieR.ELEMENT\\_MISSMATCH\\_CHECKTYPE](#) option, see there

## Usage

```
int_sts_element_dataframe(
  item_level = "item_level",
  meta_data_dataframe = "dataframe_level",
  meta_data = item_level,
  meta_data_v2,
  check_type = getOption("dataquieR.ELEMENT_MISSMATCH_CHECKTYPE",
    dataquieR.ELEMENT_MISSMATCH_CHECKTYPE_default),
  dataframe_level
)
```

## Arguments

item\_level      [data.frame](#) the data frame that contains metadata attributes of study data

meta\_data\_dataframe      [data.frame](#) the data frame that contains the metadata for the data frame level

meta\_data      [data.frame](#) old name for item\_level

meta\_data\_v2      [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify meta\_data\_v2.

check\_type      [enum](#) none | exact | subset\_u | subset\_m. See [dataquieR.ELEMENT\\_MISSMATCH\\_CHECKTYPE](#)

dataframe\_level      [data.frame](#) alias for meta\_data\_dataframe

**Details**[Indicator](#)**Value**[list](#) with names lots:

- DataFrameData: data frame with the unexpected elements check results.
- DataFrameTable: [data.frame](#) table with all errors, used for the data quality report: - PCT\_int\_sts\_element: Percentage of element mismatches - NUM\_int\_sts\_element: Number of element mismatches - resp\_vars: affected element names

**Examples**

```
## Not run:
prep_load_workbook_like_file("~/tmp/df_level_test.xlsx")
meta_data_dataframe <- "dataframe_level"
meta_data <- "item_level"

## End(Not run)
```

---

int\_sts\_element\_segment

*Checks for element set*


---

**Description**

Depends on dataquieR.ELEMENT\_MISSMATCH\_CHECKTYPE option, see there – # TODO: Rind out, how to document and link it here using Roxygen.

**Usage**

```
int_sts_element_segment(
  study_data,
  item_level = "item_level",
  label_col,
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

study_data	<a href="#">data.frame</a> the data frame that contains the measurements, mandatory.
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.

**Details**[Indicator](#)**Value**a [list](#) with

- SegmentData: data frame with the unexpected elements check results. - Segment: name of the corresponding segment, if applicable, ALL otherwise
- SegmentTable: data frame with the unexpected elements check results, used for the data quality report. - Segment: name of the corresponding segment, if applicable, ALL otherwise

**Examples**

```
## Not run:
study_data <- cars
meta_data <- dataquieR::prep_create_meta(VAR_NAMES = c("speedx", "distx"),
  DATA_TYPE = c("integer", "integer"), MISSING_LIST = "|", JUMP_LIST = "|",
  STUDY_SEGMENT = c("Intro", "Ex"))
options(dataquieR.ELEMENT_MISMATCH_CHECKTYPE = "none")
int_sts_element_segment(study_data, meta_data)
options(dataquieR.ELEMENT_MISMATCH_CHECKTYPE = "exact")
int_sts_element_segment(study_data, meta_data)
study_data <- cars
meta_data <- dataquieR::prep_create_meta(VAR_NAMES = c("speedx", "distx"),
  DATA_TYPE = c("integer", "integer"), MISSING_LIST = "|", JUMP_LIST = "|",
  STUDY_SEGMENT = c("Intro", "Intro"))
options(dataquieR.ELEMENT_MISMATCH_CHECKTYPE = "none")
int_sts_element_segment(study_data, meta_data)
options(dataquieR.ELEMENT_MISMATCH_CHECKTYPE = "exact")
int_sts_element_segment(study_data, meta_data)
study_data <- cars
meta_data <- dataquieR::prep_create_meta(VAR_NAMES = c("speed", "distx"),
  DATA_TYPE = c("integer", "integer"), MISSING_LIST = "|", JUMP_LIST = "|",
  STUDY_SEGMENT = c("Intro", "Intro"))
options(dataquieR.ELEMENT_MISMATCH_CHECKTYPE = "none")
int_sts_element_segment(study_data, meta_data)
options(dataquieR.ELEMENT_MISMATCH_CHECKTYPE = "exact")
int_sts_element_segment(study_data, meta_data)

## End(Not run)
```

---

`int_unexp_elements`*Check for unexpected data element count*

---

**Description**

This function contrasts the expected element number in each study in the metadata with the actual element number in each study data frame.

[Indicator](#)

**Usage**

```
int_unexp_elements(
  identifier_name_list,
  data_element_count,
  meta_data_dataframe = "dataframe_level",
  meta_data_v2,
  dataframe_level
)
```

**Arguments**

`identifier_name_list` **character** a character vector indicating the name of each study data frame, mandatory.

`data_element_count` **integer** an integer vector with the number of expected data elements, mandatory.

`meta_data_dataframe` **data.frame** the data frame that contains the metadata for the data frame level

`meta_data_v2` **character** path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

`dataframe_level` **data.frame** alias for `meta_data_dataframe`

**Value**

a **list** with

- **DataframeData**: data frame with the results of the quality check for unexpected data elements
- **DataframeTable**: data frame with selected unexpected data elements check results, used for the data quality report.

---

`int_unexp_records_dataframe`

*Check for unexpected data record count at the data frame level*

---

**Description**

This function contrasts the expected record number in each study in the metadata with the actual record number in each study data frame.

**Indicator**

**Usage**

```
int_unexp_records_dataframe(
  identifier_name_list,
  data_record_count,
  meta_data_dataframe = "dataframe_level",
  meta_data_v2,
  dataframe_level
)
```

**Arguments**

`identifier_name_list` **character** a character vector indicating the name of each study data frame, mandatory.

`data_record_count` **integer** an integer vector with the number of expected data records per study data frame, mandatory.

`meta_data_dataframe` **data.frame** the data frame that contains the metadata for the data frame level

`meta_data_v2` **character** path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

`dataframe_level` **data.frame** alias for `meta_data_dataframe`

**Value**

a **list** with

- `DataframeData`: data frame with the results of the quality check for unexpected data elements
- `DataframeTable`: data frame with selected unexpected data elements check results, used for the data quality report.

---

`int_unexp_records_segment`

*Check for unexpected data record count within segments*

---

**Description**

This function contrasts the expected record number in each study segment in the metadata with the actual record number in each segment data frame.

**Indicator**

**Usage**

```
int_unexp_records_segment(
  study_segment,
  study_data,
  label_col,
  item_level = "item_level",
  data_record_count,
  meta_data = item_level,
  meta_data_segment = "segment_level",
  meta_data_v2,
  segment_level
)
```

**Arguments**

`study_segment` **character** a character vector indicating the name of each study data frame, mandatory.

`study_data` **data.frame** the data frame that contains the measurements

`label_col` **variable attribute** the name of the column in the metadata with labels of variables

`item_level` **data.frame** the data frame that contains metadata attributes of study data

`data_record_count` **integer** an integer vector with the number of expected data records, mandatory.

`meta_data` **data.frame** old name for `item_level`

`meta_data_segment` **data.frame** – optional: Segment level metadata

`meta_data_v2` **character** path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

`segment_level` **data.frame** alias for `meta_data_segment`

**Details**

The current implementation does not take into account jump or missing codes, the function is rather based on checking whether NAs are present in the study data

**Value**

a **list** with

- `SegmentData`: data frame with the results of the quality check for unexpected data elements
- `SegmentTable`: data frame with selected unexpected data elements check results, used for the data quality report.

---

int\_unexp\_records\_set *Check for unexpected data record set*

---

## Description

This function tests that the identifiers match a provided record set. It is possible to check for unexpected data record sets by study segments or to consider only selected segments.

### Indicator

## Usage

```
int_unexp_records_set(
  level = c("dataframe", "segment"),
  study_data,
  item_level = "item_level",
  label_col,
  meta_data = item_level,
  meta_data_v2,
  ...
)
```

## Arguments

level	<b>character</b> a character vector indicating whether the assessment should be conducted at the study level (level = "dataframe") or at the segment level (level = "segment").
study_data	<b>data.frame</b> the data frame that contains the measurements
item_level	<b>data.frame</b> the data frame that contains metadata attributes of study data
label_col	<b>variable attribute</b> the name of the column in the metadata with labels of variables
meta_data	<b>data.frame</b> old name for item_level
meta_data_v2	<b>character</b> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
...	Depending on level, passed to either <code>util_int_unexp_records_set_segment</code> or <code>util_int_unexp_records_set_dataframe</code>

## Value

a **list**. Depending on level, see `util_int_unexp_records_set_segment` or `util_int_unexp_records_set_dataframe` for a description of the outputs.

---

IRV *Cross-item level metadata attribute name*

---

### Description

TODO

### See Also

[meta\\_data\\_cross](#)

Other [meta\\_data\\_cross](#): [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

Other SSI: [COMPUTED\\_VARIABLE\\_ROLES](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [RELCOMPL\\_SPEED](#), [RESPT\\_PER\\_ITEM](#), [TOTRESPT](#)

---

MAHALANOBIS\_THRESHOLD *Cross-item level metadata attribute name*

---

### Description

Select, whether to compute [acc\\_mahalanobis](#).

### Usage

MAHALANOBIS\_THRESHOLD

### Format

An object of class character of length 1.

### Details

You can leave the cell empty, then the depends on the setting of the option [dataquieR.MULTIVARIATE\\_OUTLIER\\_CHECK](#). If this column is missing, all this is the same as having all cells empty and [dataquieR.MULTIVARIATE\\_OUTLIER\\_CHECK](#) set to "auto".

See also [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#).

**See Also**

[meta\\_data\\_cross](#)

Other meta\_data\_cross: ASSOCIATION\_DIRECTION, ASSOCIATION\_FORM, ASSOCIATION\_METRIC, ASSOCIATION\_RANGE, CHECK\_ID, CHECK\_LABEL, COMPUTED\_VARIABLE\_ROLES, CONTRADICTION\_TERM, CONTRADICTION\_TYPE, DATA\_PREPARATION, GOLDSTANDARD, IRV, MAXIMUM\_LONG\_STRING, MISS\_RESP, MULTIVARIATE\_OUTLIER\_CHECK, MULTIVARIATE\_OUTLIER\_CHECKTYPE, RELCOMPL\_SPEED, REL\_VAL, RESPT\_PER\_ITEM, SCALE\_ACRONYM, SCALE\_NAME, TOTRESPT, VARIABLE\_LIST, VARIABLE\_LIST\_ORDER, meta\_data\_computation, meta\_data\_cross

---

MAXIMUM_LONG_STRING	<i>Cross-item level metadata attribute name</i>
---------------------	---

---

**Description**

TODO

**See Also**

[meta\\_data\\_cross](#)

Other meta\_data\_cross: ASSOCIATION\_DIRECTION, ASSOCIATION\_FORM, ASSOCIATION\_METRIC, ASSOCIATION\_RANGE, CHECK\_ID, CHECK\_LABEL, COMPUTED\_VARIABLE\_ROLES, CONTRADICTION\_TERM, CONTRADICTION\_TYPE, DATA\_PREPARATION, GOLDSTANDARD, IRV, MAHALANOBIS\_THRESHOLD, MISS\_RESP, MULTIVARIATE\_OUTLIER\_CHECK, MULTIVARIATE\_OUTLIER\_CHECKTYPE, RELCOMPL\_SPEED, REL\_VAL, RESPT\_PER\_ITEM, SCALE\_ACRONYM, SCALE\_NAME, TOTRESPT, VARIABLE\_LIST, VARIABLE\_LIST\_ORDER, meta\_data\_computation, meta\_data\_cross

Other SSI: COMPUTED\_VARIABLE\_ROLES, IRV, MISS\_RESP, RELCOMPL\_SPEED, RESPT\_PER\_ITEM, TOTRESPT

---

meta_data	<i>Data frame with metadata about the study data on variable level</i>
-----------	--

---

**Description**

Variable level metadata.

**See Also**

[further details on variable level metadata.](#)

[meta\\_data\\_segment](#)

[meta\\_data\\_dataframe](#)

---

meta\_data\_computation *Well known columns on the item\_computation\_level sheet*

---

### Description

Computation rules TODO

### See Also

Other meta\_data\_cross: [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_cross](#)

---

meta\_data\_cross *Well known columns on the cross-item\_level sheet*

---

### Description

Metadata describing groups of variables, e.g., for their multivariate distribution or for defining contradiction rules.

### See Also

[check\\_table](#)

[Online Documentation](#)

Other meta\_data\_cross: [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#)

---

meta\_data\_dataframe *Well known columns on the meta\_data\_dataframe sheet*

---

### Description

Metadata describing data delivered on one data frame/table sheet, e.g., a full questionnaire, not its items.

---

meta_data_segment	<i>Well known columns on the meta_data_segment sheet</i>
-------------------	--

---

**Description**

Metadata describing study segments, e.g., a full questionnaire, not its items.

---

MISS_RESP	<i>Cross-item level metadata attribute name</i>
-----------	---

---

**Description**

TODO

**See Also**

[meta\\_data\\_cross](#)

Other meta\_data\_cross: ASSOCIATION\_DIRECTION, ASSOCIATION\_FORM, ASSOCIATION\_METRIC, ASSOCIATION\_RANGE, CHECK\_ID, CHECK\_LABEL, COMPUTED\_VARIABLE\_ROLES, CONTRADICTION\_TERM, CONTRADICTION\_TYPE, DATA\_PREPARATION, GOLDSTANDARD, IRV, MAHALANOBIS\_THRESHOLD, MAXIMUM\_LONG\_STRING, MULTIVARIATE\_OUTLIER\_CHECK, MULTIVARIATE\_OUTLIER\_CHECKTYPE, RELCOMPL\_SPEED, REL\_VAL, RESPT\_PER\_ITEM, SCALE\_ACRONYM, SCALE\_NAME, TOTRESPT, VARIABLE\_LIST, VARIABLE\_LIST\_ORDER, meta\_data\_computation, meta\_data\_cross

Other SSI: COMPUTED\_VARIABLE\_ROLES, IRV, MAXIMUM\_LONG\_STRING, RELCOMPL\_SPEED, RESPT\_PER\_ITEM, TOTRESPT

---

MULTIVARIATE_OUTLIER_CHECK	<i>Cross-item level metadata attribute name</i>
----------------------------	---

---

**Description**

Select, whether to compute [acc\\_multivariate\\_outlier](#).

**Usage**

MULTIVARIATE\_OUTLIER\_CHECK

**Format**

An object of class character of length 1.

**Details**

You can leave the cell empty, then the depends on the setting of the option `dataquieR.MULTIVARIATE_OUTLIER_CHECKTYPE`. If this column is missing, all this is the same as having all cells empty and `dataquieR.MULTIVARIATE_OUTLIER_CHECKTYPE` set to "auto".

See also [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#).

**See Also**

[meta\\_data\\_cross](#)

Other `meta_data_cross`: [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

---

MULTIVARIATE\_OUTLIER\_CHECKTYPE

*Cross-item level metadata attribute name*

---

**Description**

Select, which outlier criteria to compute, see [acc\\_multivariate\\_outlier](#).

**Usage**

MULTIVARIATE\_OUTLIER\_CHECKTYPE

**Format**

An object of class character of length 1.

**Details**

You can leave the cell empty, then, all checks will apply. If you enter a set of methods, the maximum for [N\\_RULES](#) changes. See also [UNIVARIATE\\_OUTLIER\\_CHECKTYPE](#).

**See Also**

[meta\\_data\\_cross](#)

Other `meta_data_cross`: [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

---

nres                      *return the number of result slots in a report*

---

**Description**

return the number of result slots in a report

**Usage**

nres(x)

**Arguments**

x                      the dataquieR report (v2.0)

**Value**

the number of used result slots

---

pipeline\_recursive\_result  
*Convert a pipeline result data frame to named encapsulated lists*

---

**Description**

Deprecated

**Usage**

pipeline\_recursive\_result(...)

**Arguments**

...                      Deprecated

**Value**

Deprecated

---

pipeline_vectorized	<i>Call (nearly) one "Accuracy" function with many parameterizations at once automatically</i>
---------------------	--

---

**Description**

Deprecated

**Usage**

```
pipeline_vectorized(...)
```

**Arguments**

...           Deprecated

**Value**

Deprecated

---

plot.dataquieR_summary	<i>Plot a dataquieR summary</i>
------------------------	---------------------------------

---

**Description**

Plot a dataquieR summary

**Usage**

```
## S3 method for class 'dataquieR_summary'  
plot(  
  x,  
  y,  
  ...,  
  filter,  
  dont_plot = FALSE,  
  stratify_by,  
  vars_to_include = "study",  
  disable_plotly = FALSE  
)
```

**Arguments**

x	the dataquieR summary, see <a href="#">summary()</a> and <a href="#">dq_report2()</a>
y	not yet used
...	not yet used
filter	if given, this filters the summary, e.g., <code>filter = call_names == "com_qualified_item_missingness"</code>
dont_plot	suppress the actual plotting, just return a printable object derived from x
stratify_by	column to stratify the summary, may be one string.
vars_to_include	"study", "ssi" or <code>c("study", "ssi")</code> . variables to include
disable_plotly	<b>logical</b> do not use plotly, even if installed

**Value**

invisible html object

---

```
prep_acc_distributions_with_ecdf
```

*Utility function to plot a combined figure for distribution checks*

---

**Description**

Data quality indicator checks "Unexpected location" with histograms and plots of empirical cumulative distributions for the subgroups.

**Usage**

```
prep_acc_distributions_with_ecdf(
  resp_vars = NULL,
  group_vars = NULL,
  study_data,
  label_col,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2,
  n_group_max = getOption("dataquieR.max_group_var_levels_in_plot",
    dataquieR.max_group_var_levels_in_plot_default),
  n_obs_per_group_min = getOption("dataquieR.min_obs_per_group_var_in_plot",
    dataquieR.min_obs_per_group_var_in_plot_default)
)
```

**Arguments**

resp_vars	<a href="#">variable list</a> the name of the measurement variable
group_vars	<a href="#">variable list</a> the name of the observer, device or reader variable
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
n_group_max	maximum number of categories to be displayed individually for the grouping variable (group_vars, devices / examiners)
n_obs_per_group_min	minimum number of data points per group to create a graph for an individual category of the group_vars variable

**Value**

A SummaryPlot.

---

```
prep_add_cause_label_df
```

*Convert missing codes in metadata format v1.0 and a missing-cause-table to v2.0 missing list / jump list assignments*

---

**Description**

The function has two working modes. If `replace_meta_data` is TRUE, by default, if `cause_label_df` contains a column named `resp_vars`, then the missing/jump codes in `meta_data[, c(MISSING_CODES, JUMP_CODES)]` will be overwritten, otherwise, it will be labeled using the `cause_label_df`.

**Usage**

```
prep_add_cause_label_df(
  item_level = "item_level",
  cause_label_df,
  label_col = VAR_NAMES,
  assume_consistent_codes = TRUE,
  replace_meta_data = ("resp_vars" %in% colnames(cause_label_df)),
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
cause_label_df	<a href="#">data.frame</a> missing code table. If missing codes have labels the respective data frame can be specified here, see <a href="#">cause_label_df</a>
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
assume_consistent_codes	<a href="#">logical</a> if TRUE and no labels are given and the same missing/jump code is used for more than one variable, the labels assigned for this code will be the same for all variables.
replace_meta_data	<a href="#">logical</a> if TRUE, ignore existing missing codes and jump codes and replace them with data from the cause_label_df. Otherwise, copy the labels from cause_label_df to the existing code columns.
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.

**Details**

If a column resp\_vars exists, then rows with a value in resp\_vars will only be used for the corresponding variable.

**Value**

[data.frame](#) updated metadata including all the code labels in missing/jump lists

**See Also**

[prep\\_extract\\_cause\\_label\\_df](#)

---

```
prep_add_computed_variables
```

*Insert missing codes for NAs based on rules*

---

**Description**

Insert missing codes for NAs based on rules

**Usage**

```
prep_add_computed_variables(
  study_data,
  meta_data,
  label_col,
  rules,
  use_value_labels
)
```

**Arguments**

study_data	<a href="#">data.frame</a> the data frame that contains the measurements
meta_data	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
rules	<a href="#">data.frame</a> with the columns: <ul style="list-style-type: none"> <li>• VAR_NAMES: <a href="#">VAR_NAMES</a> of the variable to compute</li> <li>• COMPUTATION_RULE: A rule in REDcap style (see, e.g., <a href="#">REDCap help</a>, <a href="#">REDCap how-to</a>), and <a href="#">REDCap branching logic</a> that defines, how to compute the new values</li> </ul>
use_value_labels	<a href="#">logical</a> In rules for factors, use the value labels, not the codes. Defaults to TRUE, if any VALUE_LABELS are given in the metadata.

**Value**

a list with the entry:

- ModifiedStudyData: Study data with the new variables

**Examples**

```
## Not run:
study_data <- prep_get_data_frame("ship")
prep_load_workbook_like_file("ship_meta_v2")
meta_data <- prep_get_data_frame("item_level")
rules <- tibble::tribble(
  ~VAR_NAMES, ~RULE,
  "BMI", '[BODY_WEIGHT_0]/(([BODY_HEIGHT_0]/100)^2)',
  "R", '[WAIST_CIRC_0]/2/[pi]', # in m^3
  "VOL_EST", '[pi]*([WAIST_CIRC_0]/2/[pi])^2*[BODY_HEIGHT_0] / 1000', # in l
)
r <- prep_add_computed_variables(study_data, meta_data,
  label_col = "LABEL", rules, use_value_labels = FALSE)

## End(Not run)
```

---

prep\_add\_data\_frames *Add data frames to the pre-loaded / cache data frame environment*

---

**Description**

These can be referred to by their names, then, wherever dataquieR expects a [data.frame](#) – just pass a character instead. If this character is not found, dataquieR would additionally look for files with the name and for URLs. You can also refer to specific sheets of a workbook or specific object from an RData by appending a pipe symbol and its name. A second pipe symbol allows to extract certain columns from such sheets (but they will remain data frames).

**Usage**

```
prep_add_data_frames(..., data_frame_list = list())
```

**Arguments**

... data frames, if passed with names, these will be the names of these tables in the data frame environment. If not, then the names in the calling environment will be used.

data\_frame\_list a named list with data frames. Also these will be added and names will be handled as for the ... argument.

**Value**

`data.frame` invisible(the cache environment)

**See Also**

[prep\\_load\\_workbook\\_like\\_file](#)

[prep\\_get\\_data\\_frame](#)

Other data-frame-cache: [prep\\_get\\_data\\_frame\(\)](#), [prep\\_list\\_dataframes\(\)](#), [prep\\_load\\_folder\\_with\\_metadata\(\)](#), [prep\\_load\\_workbook\\_like\\_file\(\)](#), [prep\\_purge\\_data\\_frame\\_cache\(\)](#), [prep\\_remove\\_from\\_cache\(\)](#)

---

```
prep_add_missing_codes
```

*Insert missing codes for NAs based on rules*

---

**Description**

Insert missing codes for NAs based on rules

**Usage**

```
prep_add_missing_codes(
  resp_vars,
  study_data,
  meta_data_v2,
  item_level = "item_level",
  label_col,
  rules,
  use_value_labels,
  overwrite = FALSE,
  meta_data = item_level
)
```

**Arguments**

resp_vars	<a href="#">variable list</a> the name of the measurement variables to be modified, all from rules, if omitted
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
label_col	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
rules	<a href="#">data.frame</a> with the columns: <ul style="list-style-type: none"> <li>• resp_vars: Variable, whose NA-values should be replaced by jump codes</li> <li>• CODE_CLASS: Either MISSING or JUMP: Is the currently described case an expected missing value (JUMP) or not (MISSING)</li> <li>• CODE_VALUE: The jump code or missing code</li> <li>• CODE_LABEL: A label describing the reason for the missing value</li> <li>• RULE: A rule in REDcap style (see, e.g., <a href="#">REDcap help</a>, <a href="#">REDcap how-to</a>), and <a href="#">REDcap branching logic</a> that describes cases for the missing</li> </ul>
use_value_labels	<a href="#">logical</a> In rules for factors, use the value labels, not the codes. Defaults to TRUE, if any VALUE_LABELS are given in the metadata.
overwrite	<a href="#">logical</a> Also insert missing codes, if the values are not NA
meta_data	<a href="#">data.frame</a> old name for item_level attributes of study data

**Value**

a list with the entries:

- ModifiedStudyData: Study data with NAs replaced by the CODE\_VALUE
- ModifiedMetaData: Metadata having the new codes amended in the columns JUMP\_LIST or MISSING\_LIST, respectively

---

```
prep_add_to_meta
```

*Support function to augment metadata during data quality reporting*

---

**Description**

adds an annotation to static metadata

**Usage**

```

prep_add_to_meta(
  VAR_NAMES,
  DATA_TYPE,
  LABEL,
  VALUE_LABELS,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2,
  ...
)

```

**Arguments**

VAR_NAMES	<a href="#">character</a> Names of the Variables to add
DATA_TYPE	<a href="#">character</a> Data type for the added variables
LABEL	<a href="#">character</a> Labels for these variables
VALUE_LABELS	<a href="#">character</a> Value labels for the values of the variables as usually pipe separated and assigned with =: 1 = male   2 = female
item_level	<a href="#">data.frame</a> the metadata to extend
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
...	Further defined variable attributes, see <a href="#">prep_create_meta</a>

**Details**

Add metadata e.g. of transformed/new variable This function is not yet considered stable, but we already export it, because it could help. Therefore, we have some inconsistencies in the formals still.

**Value**

a data frame with amended metadata.

---

prep_apply_coding	<i>Re-Code labels with their respective codes according to the meta_data</i>
-------------------	--

---

**Description**

Re-Code labels with their respective codes according to the meta\_data

**Usage**

```
prep_apply_coding(  
  study_data,  
  meta_data_v2,  
  item_level = "item_level",  
  meta_data = item_level  
)
```

**Arguments**

study_data	<a href="#">data.frame</a> the data frame that contains the measurements
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data	<a href="#">data.frame</a> old name for item_level

**Value**

[data.frame](#) modified study data with labels replaced by the codes

---

prep\_check\_for\_dataquieR\_updates

*Check for package updates*

---

**Description**

Check for package updates

**Usage**

```
prep_check_for_dataquieR_updates(  
  beta = FALSE,  
  deps = TRUE,  
  ask = interactive()  
)
```

**Arguments**

beta	<a href="#">logical</a> check for beta version too
deps	<a href="#">logical</a> check for missing (optional) dependencies
ask	<a href="#">logical</a> ask for updates

**Value**

invisible(NULL)

---

```
prep_check_meta_data_dataframe
```

*Verify and normalize metadata on data frame level*

---

## Description

if possible, mismatching data types are converted ("true" becomes TRUE)

## Usage

```
prep_check_meta_data_dataframe(  
  meta_data_dataframe = "dataframe_level",  
  meta_data_v2,  
  dataframe_level  
)
```

## Arguments

```
meta_data_dataframe      data.frame data frame or path/url of a metadata sheet for the data frame level  
meta_data_v2            character path to workbook like metadata file, see prep\_load\_workbook\_like\_file  
                        for details. ALL LOADED DATAFRAMES WILL BE PURGED, using  
                        prep\_purge\_data\_frame\_cache, if you specify meta_data_v2.  
dataframe_level        data.frame alias for meta_data_dataframe
```

## Details

missing columns are added, filled with NA, if this is valid, i.e., n.a. for DF\_NAME as the key column

## Value

standardized metadata sheet as data frame

## Examples

```
## Not run:  
mds <- prep_check_meta_data_dataframe("ship_meta_dataframe|dataframe_level") # also converts  
print(mds)  
prep_check_meta_data_dataframe(mds)  
mds1 <- mds  
mds1$DF_RECORD_COUNT <- NULL  
print(prepare_check_meta_data_dataframe(mds1)) # fixes the missing column by NAs  
mds1 <- mds  
mds1$DF_UNIQUE_ROWS[[2]] <- "xxx" # not convertible  
# print(prepare_check_meta_data_dataframe(mds1)) # fail  
mds1 <- mds  
mds1$DF_UNIQUE_ID[[2]] <- 12
```

```
# print(prepare_check_meta_data_dataframe(mds1)) # fail
## End(Not run)
```

---

```
prep_check_meta_data_segment
```

*Verify and normalize metadata on segment level*

---

## Description

if possible, mismatching data types are converted ("true" becomes TRUE)

## Usage

```
prep_check_meta_data_segment(
  meta_data_segment = "segment_level",
  meta_data_v2,
  segment_level
)
```

## Arguments

`meta_data_segment` [data.frame](#) data frame or path/url of a metadata sheet for the segment level

`meta_data_v2` [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

`segment_level` [data.frame](#) alias for `meta_data_segment`

## Details

missing columns are added, filled with NA, if this is valid, i.e., n.a. for STUDY\_SEGMENT as the key column

## Value

standardized metadata sheet as data frame

## Examples

```
## Not run:
mds <- prep_check_meta_data_segment("ship_meta_v2|segment_level") # also converts
print(mds)
prep_check_meta_data_segment(mds)
mds1 <- mds
mds1$SEGMENT_RECORD_COUNT <- NULL
print(prepare_check_meta_data_segment(mds1)) # fixes the missing column by NAs
mds1 <- mds
```

```

mds1$SEGMENT_UNIQUE_ROWS[[2]] <- "xxx" # not convertible
# print(prepare_check_meta_data_segment(mds1)) # fail

## End(Not run)

```

---

prep\_check\_meta\_names *Checks the validity of metadata w.r.t. the provided column names*

---

## Description

This function verifies, if a data frame complies to metadata conventions and provides a given richness of meta information as specified by level.

## Usage

```

prep_check_meta_names(
  item_level = "item_level",
  level,
  character.only = FALSE,
  meta_data = item_level,
  meta_data_v2
)

```

## Arguments

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`level` [enum](#) level of requirement (see also [VARATT\\_REQUIRE\\_LEVELS](#)). set to NULL to deactivate the check of richness.

`character.only` [logical](#) a logical indicating whether level can be assumed to be character strings.

`meta_data` [data.frame](#) old name for `item_level`

`meta_data_v2` [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

## Details

Note, that only the given level is checked despite, levels are somehow hierarchical.

## Value

a logical with:

- `invisible(TRUE)`. In case of problems with the metadata, a condition is raised (`stop()`).

**Examples**

```

## Not run:
prep_check_meta_names(data.frame(VAR_NAMES = 1, DATA_TYPE = 2,
                                MISSING_LIST = 3))

prep_check_meta_names(
  data.frame(
    VAR_NAMES = 1, DATA_TYPE = 2, MISSING_LIST = 3,
    LABEL = "LABEL", VALUE_LABELS = "VALUE_LABELS",
    JUMP_LIST = "JUMP_LIST", HARD_LIMITS = "HARD_LIMITS",
    GROUP_VAR_OBSERVER = "GROUP_VAR_OBSERVER",
    GROUP_VAR_DEVICE = "GROUP_VAR_DEVICE",
    TIME_VAR = "TIME_VAR",
    PART_VAR = "PART_VAR",
    STUDY_SEGMENT = "STUDY_SEGMENT",
    LOCATION_RANGE = "LOCATION_RANGE",
    LOCATION_METRIC = "LOCATION_METRIC",
    PROPORTION_RANGE = "PROPORTION_RANGE",
    MISSING_LIST_TABLE = "MISSING_LIST_TABLE",
    CO_VARS = "CO_VARS",
    LONG_LABEL = "LONG_LABEL"
  ),
  RECOMMENDED
)

prep_check_meta_names(
  data.frame(
    VAR_NAMES = 1, DATA_TYPE = 2, MISSING_LIST = 3,
    LABEL = "LABEL", VALUE_LABELS = "VALUE_LABELS",
    JUMP_LIST = "JUMP_LIST", HARD_LIMITS = "HARD_LIMITS",
    GROUP_VAR_OBSERVER = "GROUP_VAR_OBSERVER",
    GROUP_VAR_DEVICE = "GROUP_VAR_DEVICE",
    TIME_VAR = "TIME_VAR",
    PART_VAR = "PART_VAR",
    STUDY_SEGMENT = "STUDY_SEGMENT",
    LOCATION_RANGE = "LOCATION_RANGE",
    LOCATION_METRIC = "LOCATION_METRIC",
    PROPORTION_RANGE = "PROPORTION_RANGE",
    DETECTION_LIMITS = "DETECTION_LIMITS", SOFT_LIMITS = "SOFT_LIMITS",
    CONTRADICTIONS = "CONTRADICTIONS", DISTRIBUTION = "DISTRIBUTION",
    DECIMALS = "DECIMALS", VARIABLE_ROLE = "VARIABLE_ROLE",
    DATA_ENTRY_TYPE = "DATA_ENTRY_TYPE",
    CO_VARS = "CO_VARS",
    END_DIGIT_CHECK = "END_DIGIT_CHECK",
    VARIABLE_ORDER = "VARIABLE_ORDER", LONG_LABEL =
      "LONG_LABEL", recode = "recode",
    MISSING_LIST_TABLE = "MISSING_LIST_TABLE"
  ),
  OPTIONAL
)

# Next one will fail

```

```

try(
  prep_check_meta_names(data.frame(VAR_NAMES = 1, DATA_TYPE = 2,
    MISSING_LIST = 3), TECHNICAL)
)

## End(Not run)

```

---

```

prep_clean_labels      Support function to scan variable labels for applicability

```

---

### Description

Adjust labels in `meta_data` to be valid variable names in formulas for diverse r functions, such as `glm` or `lme4::lmer`.

### Usage

```

prep_clean_labels(
  label_col,
  item_level = "item_level",
  no_dups = FALSE,
  meta_data = item_level,
  meta_data_v2
)

```

### Arguments

<code>label_col</code>	<b>character</b> label attribute to adjust or character vector to adjust, depending on <code>meta_data</code> argument is given or missing.
<code>item_level</code>	<b>data.frame</b> metadata data frame: If <code>label_col</code> is a label attribute to adjust, this is the metadata table to process on. If missing, <code>label_col</code> must be a character vector with values to adjust.
<code>no_dups</code>	<b>logical</b> disallow duplicates in input or output vectors of the function, then, <code>prep_clean_labels</code> would call <code>stop()</code> on duplicated labels.
<code>meta_data</code>	<b>data.frame</b> old name for <code>item_level</code>
<code>meta_data_v2</code>	<b>character</b> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify <code>meta_data_v2</code> .

### Details

Hint: The following is still true, but the functions should be capable of doing potentially needed fixes on-the-fly automatically, so likely you will not need this function any more.

Currently, labels as given by `label_col` arguments in the most functions are directly used in formula, so that they become natural part of the outputs, but different models expect differently strict syntax for such formulas, especially for valid variable names. `prep_clean_labels` removes all

potentially inadmissible characters from variable names (no guarantee, that some exotic model still rejects the names, but minimizing the number of exotic characters). However, variable names are modified, may become unreadable or indistinguishable from other variable names. For the latter case, a stop call is possible, controlled by the `no_dups` argument.

A warning is emitted, if modifications were necessary.

## Value

a data.frame with:

- if `meta_data` is set, a list with:
  - modified `meta_data[, label_col]` column
- if `meta_data` is not set, adjusted labels that then were directly given in `label_col`

## Examples

```
## Not run:
meta_data1 <- data.frame(
  LABEL =
    c(
      "syst. Blood pressure (mmHg) 1",
      "1st heart frequency in MHz",
      "body surface (\u033A1)"
    )
)
print(meta_data1)
print(prepare_clean_labels(meta_data1$LABEL))
meta_data1 <- prepare_clean_labels("LABEL", meta_data1)
print(meta_data1)

## End(Not run)
```

---

```
prep_combine_report_summaries
```

*Combine two report summaries*

---

## Description

Combine two report summaries

## Usage

```
prep_combine_report_summaries(..., summaries_list, amend_segment_names = FALSE)
```

**Arguments**

... objects returned by [prep\\_extract\\_summary](#)  
 summaries\_list if given, [list](#) of objects returned by [prep\\_extract\\_summary](#)  
 amend\_segment\_names [logical](#) use names of the summaries\_list and argument names as segment prefixes

**Value**

combined summaries

**See Also**

Other summary\_functions: [prep\\_extract\\_classes\\_by\\_functions\(\)](#), [prep\\_extract\\_summary\(\)](#), [prep\\_extract\\_summary.dataquieR\\_result\(\)](#), [prep\\_extract\\_summary.dataquieR\\_resultset2\(\)](#), [prep\\_render\\_pie\\_chart\\_from\\_summaryclasses\\_ggplot2\(\)](#), [prep\\_render\\_pie\\_chart\\_from\\_summaryclasses\\_plot1](#), [prep\\_summary\\_to\\_classes\(\)](#)

---

prep\_compare\_meta\_with\_study  
*Verify item-level metadata*

---

**Description**

are the provided item-level meta\_data plausible given study\_data?

**Usage**

```
prep_compare_meta_with_study(
  study_data,
  label_col,
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

study\_data [data.frame](#) the data frame that contains the measurements  
 label\_col [variable attribute](#) the name of the column in the metadata with labels of variables  
 item\_level [data.frame](#) the data frame that contains metadata attributes of study data  
 meta\_data [data.frame](#) old name for item\_level  
 meta\_data\_v2 [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify meta\_data\_v2.

**Value**

an `invisible()` list with the entries.

- `pred` `data.frame` metadata predicted from `study_data`, reduced to such metadata also available in the provided metadata
- `prov` `data.frame` provided metadata, reduced to such metadata also available in the provided `study_data`
- `m1_error` `character` VAR\_NAMES of variables with potentially wrong MISSING\_LIST
- `s1_error` `character` VAR\_NAMES of variables with potentially wrong SCALE\_LEVEL
- `dt_error` `character` VAR\_NAMES of variables with potentially wrong DATA\_TYPE

---

prep_create_meta	<i>Support function to create <code>data.frames</code> of metadata</i>
------------------	--

---

**Description**

Create a metadata data frame and map names. Generally, this function only creates a `data.frame`, but using this constructor instead of calling `data.frame(..., stringsAsFactors = FALSE)`, it becomes possible, to adapt the metadata `data.frame` in later developments, e.g. if we decide to use classes for the metadata, or if certain standard names of variable attributes change. Also, a validity check is possible to implement here.

**Usage**

```
prep_create_meta(..., stringsAsFactors = FALSE, level, character.only = FALSE)
```

**Arguments**

`...` named column vectors, names will be mapped using `WELL_KNOWN_META_VARIABLE_NAMES`, if included in `WELL_KNOWN_META_VARIABLE_NAMES` can also be a data frame, then its column names will be mapped using `WELL_KNOWN_META_VARIABLE_NAMES`

`stringsAsFactors` `logical` if the argument is a list of vectors, a data frame will be created. In this case, `stringsAsFactors` controls, whether characters will be auto-converted to Factors, which defaults here always to false independent from the `default.stringsAsFactors`.

`level` `enum` level of requirement (see also `VARATT_REQUIRE_LEVELS`) set to NULL, if not a complete metadata frame is created.

`character.only` `logical` a logical indicating whether level can be assumed to be character strings.

**Details**

For now, this calls `data.frame`, but it already renames variable attributes, if they have a different name assigned in `WELL_KNOWN_META_VARIABLE_NAMES`, e.g. `WELL_KNOWN_META_VARIABLE_NAMES$RECODE` maps to `recode` in lower case.

NB: `dataquieR` exports all names from `WELL_KNOWN_META_VARIABLE_NAME` as symbols, so `RECODE` also contains "recode".

**Value**

a data frame with:

- metadata attribute names mapped and
- metadata checked using [prep\\_check\\_meta\\_names](#) and do some more verification about conventions, such as check for valid intervals in limits)

**See Also**

[WELL\\_KNOWN\\_META\\_VARIABLE\\_NAMES](#)

---

prep\_create\_meta\_data\_file

*Instantiate a new metadata file*

---

**Description**

Instantiate a new metadata file

**Usage**

```
prep_create_meta_data_file(  
  file_name,  
  study_data,  
  open = TRUE,  
  overwrite = FALSE  
)
```

**Arguments**

file_name	<a href="#">character</a> file path to write to
study_data	<a href="#">data.frame</a> optional, study data to guess metadata from
open	<a href="#">logical</a> open the file after creation
overwrite	<a href="#">logical</a> overwrite file, if exists

**Value**

invisible(NULL)

---

`prep_create_storr_factory`

*Create a factory function for storr objects for backing a [dataquieR\\_resultset2](#)*

---

### Description

Create a factory function for storr objects for backing a [dataquieR\\_resultset2](#)

### Usage

```
prep_create_storr_factory(db_dir = tempfile(), namespace = "objects")
```

### Arguments

`db_dir` [character](#) path to the directory for the back-end, if one is created on the fly.

`namespace` [character](#) namespace for the report, so that one back-end can back several reports the returned function will try to create a storr object using a temporary folder or the folder in `db_dir`, if specified. The database will either be the `storr_rds`.

### Value

storr object or NULL, if package storr is not available

---

`prep_datatype_from_data`

*Get data types from data*

---

### Description

Get data types from data

### Usage

```
prep_datatype_from_data(  
  resp_vars = colnames(study_data),  
  study_data,  
  .dont_cast_off_cols = FALSE,  
  guess_character = getOption("dataquieR.guess_character", default =  
    dataquieR.guess_character_default)  
)
```

**Arguments**

**resp\_vars** [variable](#) names of the variables to fetch the data type from the data  
**study\_data** [data.frame](#) the data frame that contains the measurements Hint: Only data frames supported, no URL or file names.  
**.dont\_cast\_off\_cols** [logical](#) internal use, only  
**guess\_character** [logical](#) guess a data type for character columns based on the values

**Value**

vector of data types

**See Also**

[prep\\_robust\\_guess\\_data\\_type\(\)](#)

**Examples**

```
## Not run:
dataquieR::prep_datatype_from_data(cars)

## End(Not run)
```

---

```
prep_deparse_assignments
```

*Convert two vectors from a code-value-table to a key-value list*

---

**Description**

Convert two vectors from a code-value-table to a key-value list

**Usage**

```
prep_deparse_assignments(
  codes,
  labels = codes,
  split_char = SPLIT_CHAR,
  mode = c("numeric_codes", "string_codes")
)
```

**Arguments**

codes	codes, numeric or dates (as default, but string codes can be enabled using the option 'mode', see below)
labels	<a href="#">character</a> labels, same length as codes
split_char	<a href="#">character</a> split character character to split code assignments
mode	<a href="#">character</a> one of two options to insist on numeric or datetime codes (default) or to allow for string codes

**Value**

a vector with assignment strings for each row of `cbind(codes, labels)`

---

`prep_dq_data_type_of` *Get the dataquieR DATA\_TYPE of x*

---

**Description**

Get the dataquieR DATA\_TYPE of x

**Usage**

```
prep_dq_data_type_of(
  x,
  guess_character = getOption("dataquieR.guess_character", default =
    dataquieR.guess_character_default)
)
```

**Arguments**

x	object to define the dataquieR data type of
guess_character	<a href="#">logical</a> guess a data type for character columns based on the values

**Value**

the dataquieR data type as listed in DATA\_TYPES

**See Also**

[DATA\\_TYPES\\_OF\\_R\\_TYPE](#)

---

```
prep_expand_codes      Expand code labels across variables
```

---

### Description

Code labels are copied from other variables, if the code is the same and the label is set only for some variables

### Usage

```
prep_expand_codes(
  item_level = "item_level",
  suppressWarnings = FALSE,
  mix_jumps_and_missings = FALSE,
  meta_data_v2,
  meta_data = item_level
)
```

### Arguments

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`suppressWarnings` [logical](#) show warnings, if labels are expanded

`mix_jumps_and_missings` [logical](#) ignore the class of the codes for label expansion, i.e., use missing code labels as jump code labels, if the values are the same.

`meta_data_v2` [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

`meta_data` [data.frame](#) old name for `item_level`

### Value

[data.frame](#) an updated metadata data frame.

### Examples

```
## Not run:
meta_data <- prep_get_data_frame("meta_data")
meta_data$JUMP_LIST[meta_data$VAR_NAMES == "v00003"] <- "99980 = NOOP"
md <- prep_expand_codes(meta_data)
md$JUMP_LIST
md$MISSING_LIST
md <- prep_expand_codes(meta_data, mix_jumps_and_missings = TRUE)
md$JUMP_LIST
md$MISSING_LIST
meta_data <- prep_get_data_frame("meta_data")
```

```

meta_data$MISSING_LIST[meta_data$VAR_NAMES == "v00003"] <- "99980 = NOOP"
md <- prep_expand_codes(meta_data)
md$JUMP_LIST
md$MISSING_LIST

## End(Not run)

```

---

```
prep_extract_cause_label_df
```

*Extract all missing/jump codes from metadata and export a cause-label-data-frame*

---

### Description

Extract all missing/jump codes from metadata and export a cause-label-data-frame

### Usage

```

prep_extract_cause_label_df(
  item_level = "item_level",
  label_col = VAR_NAMES,
  meta_data_v2,
  meta_data = item_level
)

```

### Arguments

<code>item_level</code>	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
<code>label_col</code>	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
<code>meta_data_v2</code>	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify <code>meta_data_v2</code> .
<code>meta_data</code>	<a href="#">data.frame</a> old name for <code>item_level</code>

### Value

[list](#) with the entries

- `meta_data` [data.frame](#) a data frame that contains updated metadata – you still need to add a column `MISSING_LIST_TABLE` and add the `cause_label_df` as such to the metadata cache using [prep\\_add\\_data\\_frames\(\)](#), manually.
- `cause_label_df` [data.frame](#) missing code table. If missing codes have labels the respective data frame are specified here, see [cause\\_label\\_df](#).

### See Also

[prep\\_add\\_cause\\_label\\_df](#)

prep\_extract\_classes\_by\_functions

*Extract old function based summary from data quality results*

---

### Description

Extract old function based summary from data quality results

### Usage

```
prep_extract_classes_by_functions(r)
```

### Arguments

r [dq\\_report2](#)

### Value

[data.frame](#) long format, compatible with [prep\\_summary\\_to\\_classes\(\)](#)

### See Also

Other summary\_functions: [prep\\_combine\\_report\\_summaries\(\)](#), [prep\\_extract\\_summary\(\)](#), [prep\\_extract\\_summary.dataquieR\\_resultset2\(\)](#), [prep\\_render\\_pie\\_chart\\_from\\_summaryclasses\\_ggplot2\(\)](#), [prep\\_render\\_pie\\_chart\\_from\\_summaryclasses\\_plotly\(\)](#), [prep\\_summary\\_to\\_classes\(\)](#)

---

prep\_extract\_summary *Extract summary from data quality results*

---

### Description

Generic function, currently supports [dq\\_report2](#) and [dataquieR\\_result](#)

### Usage

```
prep_extract_summary(r, ...)
```

### Arguments

r [dq\\_report2](#) or [dataquieR\\_result](#) object  
... further arguments, maybe needed for some implementations

**Value**

`list` with two slots `Data` and `Table` with `data.frames` featuring all metrics columns from the report or result in `x`, the `STUDY_SEGMENT` and the `VAR_NAMES`. In case of `Data`, the columns are formatted nicely but still with the standardized column names – use `util_translate_indicator_metrics()` to rename them nicely. In case of `Table`, just as they are.

**See Also**

Other `summary_functions`: `prep_combine_report_summaries()`, `prep_extract_classes_by_functions()`, `prep_extract_summary.dataquieR_result()`, `prep_extract_summary.dataquieR_resultset2()`, `prep_render_pie_chart_from_summaryclasses_ggplot2()`, `prep_render_pie_chart_from_summaryclasses_plotly()`, `prep_summary_to_classes()`

---

```
prep_extract_summary.dataquieR_result
```

*Extract report summary from reports*

---

**Description**

Extract report summary from reports

**Usage**

```
## S3 method for class 'dataquieR_result'
prep_extract_summary(r, ...)
```

**Arguments**

```
r           dataquieR_result a result from adq_report2 report
...         not used
```

**Value**

`list` with two slots `Data` and `Table` with `data.frames` featuring all metrics columns from the report `r`, the `STUDY_SEGMENT` and the `VAR_NAMES`. In case of `Data`, the columns are formatted nicely but still with the standardized column names – use `util_translate_indicator_metrics()` to rename them nicely. In case of `Table`, just as they are.

**See Also**

`prep_combine_report_summaries()`

Other `summary_functions`: `prep_combine_report_summaries()`, `prep_extract_classes_by_functions()`, `prep_extract_summary()`, `prep_extract_summary.dataquieR_resultset2()`, `prep_render_pie_chart_from_summaryclasses_ggplot2()`, `prep_render_pie_chart_from_summaryclasses_plotly()`, `prep_summary_to_classes()`

---

```
prep_extract_summary.dataquieR_resultset2
```

*Extract report summary from reports*

---

### Description

Extract report summary from reports

### Usage

```
## S3 method for class 'dataquieR_resultset2'
prep_extract_summary(r, ...)
```

### Arguments

<code>r</code>	<code>dq_report2</code> a <code>dq_report2</code> report
<code>...</code>	not used

### Value

`list` with two slots `Data` and `Table` with `data.frames` featuring all metrics columns from the report `r`, the `STUDY_SEGMENT` and the `VAR_NAMES`. In case of `Data`, the columns are formatted nicely but still with the standardized column names – use `util_translate_indicator_metrics()` to rename them nicely. In case of `Table`, just as they are.

### See Also

`prep_combine_report_summaries()`

Other `summary_functions`: `prep_combine_report_summaries()`, `prep_extract_classes_by_functions()`, `prep_extract_summary()`, `prep_extract_summary.dataquieR_result()`, `prep_render_pie_chart_from_summaryclasses_plotly()`, `prep_summary_to_classes()`

---

```
prep_fix_meta_id_dups Fix metadata duplicates
```

---

### Description

if `VAR_NAMES` have duplicates, maybe, it's because of `ID`-vars assigned to different study segments multiple times (they should be in one "intro"- segment, only), which is not the intended use of `STUDY_SEGMENT`. Naturally, they will be part of more than one data-frame, so this would also qualify for a dump duplicate, only, which can safely be removed. Only `ID`-vars are by default assumed to have such duplicates in item level metadata allowed.

**Usage**

```
prep_fix_meta_id_dups(
  meta_data_segment = "segment_level",
  meta_data_dataframe = "dataframe_level",
  item_level = "item_level",
  meta_data = item_level,
  meta_data_v2,
  segment_level,
  dataframe_level
)
```

**Arguments**

`meta_data_segment` [data.frame](#) – optional: Segment level metadata

`meta_data_dataframe` [data.frame](#) the data frame that contains the metadata for the data frame level

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`meta_data` [data.frame](#) old name for `item_level`

`meta_data_v2` [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

`segment_level` [data.frame](#) alias for `meta_data_segment`

`dataframe_level` [data.frame](#) alias for `meta_data_dataframe`

**Value**

[meta\\_data](#)

**Examples**

```
## Not run:
i1 <- prep_get_data_frame("item_level")
i1 <- rbind(i1, i1)
i1$STUDY_SEGMENT[2] <- "X"
i12 <- prep_fix_meta_id_dups(meta_data_v2 = "meta_data_v2", item_level = i1)
i12$STUDY_SEGMENT
i1$STUDY_SEGMENT[3] <- "X"
i13 <- prep_fix_meta_id_dups(meta_data_v2 = "meta_data_v2", item_level = i1)
i13$STUDY_SEGMENT

## End(Not run)
```

---

prep\_get\_data\_frame    *Read data from files/URLs*

---

### Description

data\_frame\_name can be a file path or an URL you can append a pipe and a sheet name for Excel files or object name e.g. for RData files. Numbers may also work. All file formats supported by your rio installation will work.

### Usage

```
prep_get_data_frame(  
  data_frame_name,  
  .data_frame_list = .dataframe_environment(),  
  keep_types = FALSE,  
  column_names_only = FALSE  
)
```

### Arguments

data\_frame\_name    **character** name of the data frame to read, see details

.data\_frame\_list    **environment** cache for loaded data frames

keep\_types    **logical** keep types as possibly defined in a file, if the data frame is loaded from one. set TRUE for study data.

column\_names\_only    **logical** if TRUE imports only headers (column names) of the data frame and no content (an empty data frame)

### Details

The data frames will be cached automatically, you can define an alternative environment for this using the argument .data\_frame\_list, and you can purge the cache using [prep\\_purge\\_data\\_frame\\_cache](#).

Use [prep\\_add\\_data\\_frames](#) to manually add data frames to the cache, e.g., if you have loaded them from more complex sources, before.

### Value

**data.frame** a data frame

### See Also

[prep\\_add\\_data\\_frames](#)

[prep\\_load\\_workbook\\_like\\_file](#)

Other data-frame-cache: [prep\\_add\\_data\\_frames\(\)](#), [prep\\_list\\_dataframes\(\)](#), [prep\\_load\\_folder\\_with\\_metadata\(\)](#), [prep\\_load\\_workbook\\_like\\_file\(\)](#), [prep\\_purge\\_data\\_frame\\_cache\(\)](#), [prep\\_remove\\_from\\_cache\(\)](#)

**Examples**

```
## Not run:
bl <- as.factor(prepare_data_frame(
  paste0("https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus",
        "/Projekte_RKI/COVID-19_Todesfaelle.xlsx?__blob=",
        "publicationFile|COVID_Todesfalle_BL|Bundesland"))[[1]])

n <- as.numeric(prepare_data_frame(paste0(
  "https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/",
  "Projekte_RKI/COVID-19_Todesfaelle.xlsx?__blob=",
  "publicationFile|COVID_Todesfalle_BL|Anzahl verstorbene",
  " COVID-19 Falle"))[[1]])
plot(bl, n)
# Working names would be to date (2022-10-21), e.g.:
#
# https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/ \
#   Projekte_RKI/COVID-19_Todesfaelle.xlsx?__blob=publicationFile
# https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/ \
#   Projekte_RKI/COVID-19_Todesfaelle.xlsx?__blob=publicationFile|2
# https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/ \
#   Projekte_RKI/COVID-19_Todesfaelle.xlsx?__blob=publicationFile|name
# study_data
# ship
# meta_data
# ship_meta
#
prepare_data_frame("meta_data | meta_data")

## End(Not run)
```

---

```
prep_get_labels
```

```
Fetch a label for a variable based on its purpose
```

---

**Description**

Fetch a label for a variable based on its purpose

**Usage**

```
prepare_labels(
  resp_vars,
  item_level = "item_level",
  label_col,
  max_len,
  label_class = c("SHORT", "LONG"),
  label_lang = getOption("dataquiere.lang", dataquiere.lang_default),
  resp_vars_are_var_names_only = FALSE,
  resp_vars_match_label_col_only = FALSE,
  meta_data = item_level,
```

```

    meta_data_v2,
    force_label_col = getOption("dataquieR.force_label_col",
      dataquieR.force_label_col_default)
  )

```

### Arguments

`resp_vars` **variable list** the variable names to fetch for

`item_level` **data.frame** the data frame that contains metadata attributes of study data

`label_col` **variable attribute** the name of the column in the metadata with labels of variables

`max_len` **integer** the maximum label length to return, if not possible w/o causing ambiguous labels, the labels may still be longer. For `label_class == "LONG"`, it defaults to 200, while for `label_class == "SHORT"` to 30

`label_class` **enum** SHORT | LONG. which sort of label according to the metadata model should be returned

`label_lang` **character** optional language suffix, if available in the metadata. Can be controlled by the option `dataquieR.lang`.

`resp_vars_are_var_names_only` **logical** If TRUE, do not use other labels than `VAR_NAMES` for finding `resp_vars` in `meta_data`

`resp_vars_match_label_col_only` **logical** If TRUE, do not use other labels than those, referred by `label_col` for finding `resp_vars` in `meta_data`

`meta_data` **data.frame** old name for `item_level`

`meta_data_v2` **character** path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify `meta_data_v2`.

`force_label_col` **enum** auto | FALSE | TRUE. if TRUE, always use labels according `label_col`, FALSE means use labels matching best the function's requirements, auto means FALSE, if in a `dq_report()` and TRUE, otherwise.

### Value

**character** suitable labels for each `resp_vars`, names of this vector are `VAR_NAMES`

### Examples

```

## Not run:
prep_load_workbook_like_file("meta_data_v2")
prep_get_labels("SEX_0", label_class = "SHORT", max_len = 2)

## End(Not run)

```

---

```
prep_get_study_data_segment
```

*Get data frame for a given segment*

---

## Description

Get data frame for a given segment

## Usage

```
prep_get_study_data_segment(  
  segment,  
  study_data,  
  item_level = "item_level",  
  meta_data = item_level,  
  meta_data_v2,  
  segment_level,  
  meta_data_segment = "segment_level"  
)
```

## Arguments

segment	<a href="#">character</a> name of the segment to return data for
study_data	<a href="#">data.frame</a> the data frame that contains the measurements
item_level	<a href="#">data.frame</a> the data frame that contains metadata attributes of study data
meta_data	<a href="#">data.frame</a> old name for item_level
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
segment_level	<a href="#">data.frame</a> alias for meta_data_segment
meta_data_segment	<a href="#">data.frame</a> – optional: Segment level metadata

## Value

[data.frame](#) the data for the segment

---

prep\_get\_user\_name      *Return the logged-in User's Full Name*

---

### Description

If whoami is not installed, the user name from `Sys.info()` is returned.

### Usage

```
prep_get_user_name()
```

### Details

Can be overridden by options or environment:

```
options(FULLNAME = "Stephan Struckmann")
```

```
Sys.setenv(FULLNAME = "Stephan Struckmann")
```

### Value

[character](#) the user's name

---

prep\_guess\_encoding      *Guess encoding of text or text files*

---

### Description

Guess encoding of text or text files

### Usage

```
prep_guess_encoding(x, file)
```

### Arguments

x                      [character](#) string to guess encoding for

file                    [character](#) file to guess encoding for

### Value

encoding

---

prep_link_escape	<i>Prepare a label as part of a link for RMD files</i>
------------------	--

---

**Description**

Prepare a label as part of a link for RMD files

**Usage**

```
prep_link_escape(s, html = FALSE)
```

**Arguments**

s	the label
html	prepare the label for direct HTML output instead of RMD

**Value**

the escaped label

---

prep_list_dataframes	<i>List Loaded Data Frames</i>
----------------------	--------------------------------

---

**Description**

List Loaded Data Frames

**Usage**

```
prep_list_dataframes()
```

**Value**

names of all loaded data frames

**See Also**

Other data-frame-cache: [prep\\_add\\_data\\_frames\(\)](#), [prep\\_get\\_data\\_frame\(\)](#), [prep\\_load\\_folder\\_with\\_metadata\(\)](#), [prep\\_load\\_workbook\\_like\\_file\(\)](#), [prep\\_purge\\_data\\_frame\\_cache\(\)](#), [prep\\_remove\\_from\\_cache\(\)](#)

---

prep_list_voc	<i>All valid voc: vocabularies</i>
---------------	------------------------------------

---

## Description

All valid voc: vocabularies

## Usage

```
prep_list_voc()
```

## Value

`character()` all voc: suffixes allowed for `prep_get_data_frame()`.

## Examples

```
## Not run:
prep_list_dataframes()
prep_list_voc()
prep_get_data_frame("<ICD10>")
my_voc <-
  tibble::tribble(
    ~ voc, ~ url,
    "test", "data:datasets|iris|Species+Sepal.Length")
prep_add_data_frames(`<>` = my_voc)
prep_list_dataframes()
prep_list_voc()
prep_get_data_frame("<test>")
prep_get_data_frame("<ICD10>")
my_voc <-
  tibble::tribble(
    ~ voc, ~ url,
    "ICD10", "data:datasets|iris|Species+Sepal.Length")
prep_add_data_frames(`<>` = my_voc)
prep_list_dataframes()
prep_list_voc()
prep_get_data_frame("<ICD10>")

## End(Not run)
```

---

`prep_load_folder_with_metadata`*Pre-load a folder with named (usually more than) one table(s)*

---

## Description

The original purpose of this function is to load metadata, not study data. If you want to load study data, you should keep them in a different folder, then you can call this function once for the metadata and once for the study data but this time setting `keep_types = TRUE` to avoid all data being read as `character()`.

## Usage

```
prep_load_folder_with_metadata(folder, keep_types = FALSE, ...)
```

## Arguments

<code>folder</code>	the folder name to load.
<code>keep_types</code>	<b>logical</b> keep types as possibly defined in the file. set TRUE for study data.
<code>...</code>	arguments passed to <code>list.files()</code>

## Details

Note, that once loaded to the data frame cache, a file won't be read again, except you call `prep_purge_data_frame_cache()` or `prep_remove_from_cache()`. That is, if you call this function first, and `prep_get_data_frame()` later, of if dataquieR wants to read a file, e.g., for `dq_report2()`, the file will come from the cache in the way it was initially read in (`keep_types` may thus be used inadequately).

By default, this function works not recursively, but you can tweak that by passing `...`-arguments passed through to the initially running `list.files()` function.

These can thereafter be referred to by their names only. Such files are, e.g., spreadsheet-workbooks or RData-files.

Note, that this function in contrast to `prep_get_data_frame` does neither support selecting specific sheets/columns from a file.

## Value

`invisible(the cache environment)`

## See Also

[prep\\_add\\_data\\_frames](#)

[prep\\_get\\_data\\_frame](#)

Other data-frame-cache: [prep\\_add\\_data\\_frames\(\)](#), [prep\\_get\\_data\\_frame\(\)](#), [prep\\_list\\_dataframes\(\)](#), [prep\\_load\\_workbook\\_like\\_file\(\)](#), [prep\\_purge\\_data\\_frame\\_cache\(\)](#), [prep\\_remove\\_from\\_cache\(\)](#)

---

```
prep_load_report      Load a dq_report2
```

---

**Description**

Load a dq\_report2

**Usage**

```
prep_load_report(file)
```

**Arguments**

file                    **character** the file name to load from

**Value**

**dataquieR\_resultset2** the report

---

```
prep_load_report_from_backend
      Load a report from a back-end
```

---

**Description**

Load a report from a back-end

**Usage**

```
prep_load_report_from_backend(
  namespace = "objects",
  db_dir,
  storr_factory = prep_create_storr_factory(namespace = namespace, db_dir = db_dir)
)
```

**Arguments**

namespace            the namespace to read the report's results from

db\_dir                **character** path to the directory for the back-end, if a storr\_rds or storr\_torr is used.

storr\_factory        a function returning a storr object holding the report

**Value**

**dataquieR\_resultset2** the report

**See Also**

[prep\\_create\\_storr\\_factory\(\)](#)

**Examples**

```
## Not run:
r <- dataquieR::dq_report2("study_data", meta_data_v2 = "meta_data_v2",
                          dimensions = NULL)
storr_factory <- prep_create_storr_factory()
r_storr <- prep_set_backend(r, storr_factory)
r_restorr <- prep_set_backend(r_storr, NULL)
r_loaded <- prep_load_report_from_backend(storr_factory)

## End(Not run)
```

---

prep\_load\_workbook\_like\_file

*Pre-load a file with named (usually more than) one table(s)*

---

**Description**

These can thereafter be referred to by their names only. Such files are, e.g., spreadsheet-workbooks or RData-files.

**Usage**

```
prep_load_workbook_like_file(file, keep_types = FALSE)
```

**Arguments**

`file` the file name to load.  
`keep_types` **logical** keep types as possibly defined in the file. set TRUE for study data.

**Details**

Note, that this function in contrast to [prep\\_get\\_data\\_frame](#) does neither support selecting specific sheets/columns from a file.

**Value**

invisible(the cache environment)

**See Also**

[prep\\_add\\_data\\_frames](#)

[prep\\_get\\_data\\_frame](#)

Other data-frame-cache: [prep\\_add\\_data\\_frames\(\)](#), [prep\\_get\\_data\\_frame\(\)](#), [prep\\_list\\_dataframes\(\)](#), [prep\\_load\\_folder\\_with\\_metadata\(\)](#), [prep\\_purge\\_data\\_frame\\_cache\(\)](#), [prep\\_remove\\_from\\_cache\(\)](#)

---

```
prep_map_labels      Support function to allocate labels to variables
```

---

## Description

Map variables to certain attributes, e.g. by default their labels.

## Usage

```
prep_map_labels(
  x,
  item_level = "item_level",
  to = LABEL,
  from = VAR_NAMES,
  ifnotfound,
  warn_ambiguous = FALSE,
  meta_data_v2,
  meta_data = item_level
)
```

## Arguments

x	<a href="#">character</a> variable names, character vector, see parameter from
item_level	<a href="#">data.frame</a> metadata data frame, if, as a dataquieR developer, you do not have <b>item-level-metadata</b> , you should use <code>util_map_labels()</code> instead to avoid consistency checks on for item-level meta_data.
to	<a href="#">character</a> variable attribute to map to
from	<a href="#">character</a> variable identifier to map from
ifnotfound	<a href="#">list</a> A list of values to be used if the item is not found: it will be coerced to a list if necessary.
warn_ambiguous	<a href="#">logical</a> print a warning if mapping variables from from to to produces ambiguous identifiers.
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify meta_data_v2.
meta_data	<a href="#">data.frame</a> old name for item_level

## Details

This function basically calls `colnames(study_data) <- meta_data$LABEL`, ensuring correct merging/joining of study data columns to the corresponding metadata rows, even if the orders differ. If a variable/study\_data-column name is not found in `meta_data[[from]]` (default from = VAR\_NAMES), either stop is called or, if ifnotfound has been assigned a value, that value is returned. See [mget](#), which is internally used by this function.

The function not only maps to the LABEL column, but to can be any metadata variable attribute, so the function can also be used, to get, e.g. all HARD\_LIMITS from the metadata.

**Value**

a character vector with:

- mapped values

**Examples**

```
## Not run:
meta_data <- prep_create_meta(
  VAR_NAMES = c("ID", "SEX", "AGE", "DOE"),
  LABEL = c("Pseudo-ID", "Gender", "Age", "Examination Date"),
  DATA_TYPE = c(DATA_TYPES$INTEGER, DATA_TYPES$INTEGER, DATA_TYPES$INTEGER,
                 DATA_TYPES$DATETIME),
  MISSING_LIST = ""
)
stopifnot(all(prepare_map_labels(c("AGE", "DOE"), meta_data) == c("Age",
                                                                "Examination Date")))

## End(Not run)
```

---

prep\_merge\_study\_data *Merge a list of study data frames to one (sparse) study data frame*

---

**Description**

Merge a list of study data frames to one (sparse) study data frame

**Usage**

```
prep_merge_study_data(study_data_list)
```

**Arguments**

study\_data\_list  
[list](#) the list

**Value**

[data.frame](#) study\_data

---

```
prep_meta_data_v1_to_item_level_meta_data
```

*Convert item-level metadata from v1.0 to v2.0*

---

### Description

This function is idempotent..

### Usage

```
prep_meta_data_v1_to_item_level_meta_data(  
  item_level = "item_level",  
  verbose = TRUE,  
  label_col = LABEL,  
  cause_label_df,  
  meta_data = item_level  
)
```

### Arguments

<code>item_level</code>	<a href="#">data.frame</a> the old item-level-metadata
<code>verbose</code>	<a href="#">logical</a> display all estimated decisions, defaults to TRUE, except if called in a <a href="#">dq_report2</a> pipeline.
<code>label_col</code>	<a href="#">variable attribute</a> the name of the column in the metadata with labels of variables
<code>cause_label_df</code>	<a href="#">data.frame</a> missing code table, see <a href="#">cause_label_df</a> . Optional. If this argument is given, you can add missing code tables.
<code>meta_data</code>	<a href="#">data.frame</a> old name for <code>item_level</code>

### Details

The options("dataquieR.force\_item\_specific\_missing\_codes") (default FALSE) tells the system, to always fill in `res_vars` columns to the `MISSING_LIST_TABLE`, even, if the column already exists, but is empty.

### Value

[data.frame](#) the updated metadata

---

prep\_min\_obs\_level      *Support function to identify the levels of a process variable with minimum number of observations*

---

### Description

utility function to subset data based on minimum number of observation per level

### Usage

```
prep_min_obs_level(study_data, group_vars, min_obs_in_subgroup)
```

### Arguments

study\_data      [data.frame](#) the data frame that contains the measurements  
group\_vars      [variable list](#) the name grouping variable  
min\_obs\_in\_subgroup  
                  [integer](#) optional argument if a "group\_var" is used. This argument specifies the minimum no. of observations that is required to include a subgroup (level) of the "group\_var" in the analysis. Subgroups with less observations are excluded. The default is 30.

### Details

This functions removes observations having fewer than min\_obs\_in\_subgroup distinct values in a group variable, e.g. blood pressure measurements performed by an examiner having fewer than e.g. 50 measurements done. It displays a warning, if samples/rows are removed and returns the modified study data frame.

### Value

a data frame with:

- a subsample of original data

---

prep\_open\_in\_excel      *Open a data frame in Excel*

---

### Description

Open a data frame in Excel

### Usage

```
prep_open_in_excel(dfr)
```

**Arguments**

dfr                    the data frame

**Details**

if the file cannot be read on function exit, NULL will be returned

**Value**

potentially modified data frame after dialog was closed

---

prep\_pmap                    *Support function for a parallel pmap*

---

**Description**

parallel version of purrr::pmap

**Usage**

```
prep_pmap(.l, .f, ..., cores = 0)
```

**Arguments**

.l                    [data.frame](#) with one call per line and one function argument per column

.f                    [function](#) to call with the arguments from .l

...                    additional, static arguments for calling .f

cores                    number of cpu cores to use or a (named) list with arguments for [parallelMap::parallelStart](#) or NULL, if parallel has already been started by the caller. Set to 0 to run without parallelization.

**Value**

[list](#) of results of the function calls

**Author(s)**

[Aurèle](#)  
S Struckmann

**See Also**

purrr::pmap  
[Stack Overflow post](#)

---

```
prep_prepare_dataframes
```

*Prepare and verify study data with metadata*

---

## Description

This function ensures, that a data frame `ds1` with suitable variable names `study_data` and `meta_data` exist as base [data.frames](#).

## Usage

```
prep_prepare_dataframes(
  .study_data,
  .meta_data,
  .label_col,
  .replace_hard_limits,
  .replace_missings,
  .sm_code = NULL,
  .allow_empty = FALSE,
  .adjust_data_type = TRUE,
  .amend_scale_level = TRUE,
  .apply_factor_metadata = FALSE,
  .apply_factor_metadata_inadm = FALSE,
  .internal = rlang::env_inherits(rlang::caller_env(), parent.env(environment()))
)
```

## Arguments

- `.study_data` if provided, use this data set as `study_data`
- `.meta_data` if provided, use this data set as `meta_data`
- `.label_col` if provided, use this as `label_col`
- `.replace_hard_limits`  
replace `HARD_LIMIT` violations by `NA`, defaults to `FALSE`.
- `.replace_missings`  
replace missing codes, defaults to `TRUE`
- `.sm_code` missing code for `NAs`, if they have been re-coded by `util_combine_missing_lists`
- `.allow_empty` allow `ds1` to be empty, i.e., 0 rows and/or 0 columns
- `.adjust_data_type`  
ensure that the data type of variables in the study data corresponds to their data type specified in the metadata
- `.amend_scale_level`  
ensure that `SCALE_LEVEL` is available in the item-level `meta_data`. internally used to prevent recursion, if called from [prep\\_scalelevel\\_from\\_data\\_and\\_metadata\(\)](#).
- `.apply_factor_metadata`  
[logical](#) convert categorical variables to labeled factors.

`.apply_factor_metadata_inadm` **logical** convert categorical variables to labeled factors keeping inadmissible values. Implies, that `.apply_factor_metadata` will be set to TRUE, too.

`.internal` **logical** internally called, modify caller's environment.

### Details

This function defines `ds1` and modifies `study_data` and `meta_data` in the environment of its caller (see `eval.parent`). It also defines or modifies the object `label_col` in the calling environment. Almost all functions exported by `dataquieR` call this function initially, so that aspects common to all functions live here, e.g. testing, if an argument `meta_data` has been given and features really a `data.frame`. It verifies the existence of required metadata attributes (`VARATT_REQUIRE_LEVELS`). It can also replace missing codes by NAs, and calls `prep_study2meta` to generate a minimum set of metadata from the study data on the fly (should be amended, so on-the-fly-calling is not recommended for an instructive use of `dataquieR`).

The function also detects tibbles, which are then converted to base-R `data.frames`, which are expected by `dataquieR`.

If `.internal` is TRUE, differently from the other utility function that work in their caller's environment, this function modifies objects in the calling function's environment. It defines a new object `ds1`, it modifies `study_data` and/or `meta_data` and `label_col`.

### Value

`ds1` the study data with mapped column names, `invisible()`, if not `.internal`

### See Also

`acc_margins`

### Examples

```
## Not run:
acc_test1 <- function(resp_variable, aux_variable,
                      time_variable, co_variables,
                      group_vars, study_data, meta_data) {
  prep_prepare_dataframes()
  invisible(ds1)
}
acc_test2 <- function(resp_variable, aux_variable,
                      time_variable, co_variables,
                      group_vars, study_data, meta_data, label_col) {
  ds1 <- prep_prepare_dataframes(study_data, meta_data)
  invisible(ds1)
}
environment(acc_test1) <- asNamespace("dataquieR")
# perform this inside the package (not needed for functions that have been
# integrated with the package already)

environment(acc_test2) <- asNamespace("dataquieR")
# perform this inside the package (not needed for functions that have been
```

```

# integrated with the package already)
acc_test3 <- function(resp_variable, aux_variable, time_variable,
                      co_variables, group_vars, study_data, meta_data,
                      label_col) {
  prep_prepare_dataframes()
  invisible(ds1)
}
acc_test4 <- function(resp_variable, aux_variable, time_variable,
                      co_variables, group_vars, study_data, meta_data,
                      label_col) {
  ds1 <- prep_prepare_dataframes(study_data, meta_data)
  invisible(ds1)
}
environment(acc_test3) <- asNamespace("dataquieR")
# perform this inside the package (not needed for functions that have been
# integrated with the package already)

environment(acc_test4) <- asNamespace("dataquieR")
# perform this inside the package (not needed for functions that have been
# integrated with the package already)
meta_data <- prep_get_data_frame("meta_data")
study_data <- prep_get_data_frame("study_data")
try(acc_test1())
try(acc_test2())
acc_test1(study_data = study_data)
try(acc_test1(meta_data = meta_data))
try(acc_test2(study_data = 12, meta_data = meta_data))
print(head(acc_test1(study_data = study_data, meta_data = meta_data)))
print(head(acc_test2(study_data = study_data, meta_data = meta_data)))
print(head(acc_test3(study_data = study_data, meta_data = meta_data)))
print(head(acc_test3(study_data = study_data, meta_data = meta_data,
                      label_col = LABEL)))
print(head(acc_test4(study_data = study_data, meta_data = meta_data)))
print(head(acc_test4(study_data = study_data, meta_data = meta_data,
                      label_col = LABEL)))
try(acc_test2(study_data = NULL, meta_data = meta_data))

## End(Not run)

```

---

```
prep_purge_data_frame_cache
```

*Clear data frame cache*

---

## Description

Clear data frame cache

## Usage

```
prep_purge_data_frame_cache()
```

**Value**

nothing

**See Also**

Other data-frame-cache: [prep\\_add\\_data\\_frames\(\)](#), [prep\\_get\\_data\\_frame\(\)](#), [prep\\_list\\_dataframes\(\)](#), [prep\\_load\\_folder\\_with\\_metadata\(\)](#), [prep\\_load\\_workbook\\_like\\_file\(\)](#), [prep\\_remove\\_from\\_cache\(\)](#)

---

`prep_realize_ggplot`     *Materialize a lazy ggplot*

---

**Description**

Evaluate the stored expression in its lean environment and cache the resulting ggplot object in the current R session, if enabled using the option [dataquieR.lazy\\_plots\\_cache](#).

**Usage**

```
prep_realize_ggplot(x)
```

**Arguments**

`x`                    a `dq_lazy_ggplot` object.

**Value**

A ggplot object.

---

`prep_remove_from_cache`  
*Remove a specified element from the data frame cache*

---

**Description**

Remove a specified element from the data frame cache

**Usage**

```
prep_remove_from_cache(object_to_remove)
```

**Arguments**

`object_to_remove`  
[character](#) name of the object to be removed as character string (quoted), or character vector containing the names of the objects to remove from the cache

**Value**

nothing

**See Also**

Other data-frame-cache: [prep\\_add\\_data\\_frames\(\)](#), [prep\\_get\\_data\\_frame\(\)](#), [prep\\_list\\_dataframes\(\)](#), [prep\\_load\\_folder\\_with\\_metadata\(\)](#), [prep\\_load\\_workbook\\_like\\_file\(\)](#), [prep\\_purge\\_data\\_frame\\_cache\(\)](#)

**Examples**

```
## Not run:
prep_load_workbook_like_file("meta_data_v2") #load metadata in the cache
ls(.dataframe_environment()) #get the list of dataframes in the cache

#remove cross-item_level from the cache
prep_remove_from_cache("cross-item_level")

#remove dataframe_level and expected_id from the cache
prep_remove_from_cache(c("dataframe_level", "expected_id"))

#remove missing_table and segment_level from the cache
x<- c("missing_table", "segment_level")
prep_remove_from_cache(x)

## End(Not run)
```

---

```
prep_render_pie_chart_from_summaryclasses_ggplot2
  Create a ggplot2 pie chart
```

---

**Description**

needs htmltools

**Usage**

```
prep_render_pie_chart_from_summaryclasses_ggplot2(
  data,
  meta_data = "item_level"
)
```

**Arguments**

data	data as returned by <code>prep_summary_to_classes</code> but summarized by one column (currently, we support <code>indicator_metric</code> , <code>STUDY_SEGMENT</code> , and <code>VAR_NAMES</code> )
meta_data	<a href="#">meta_data</a>

**Value**

a `htmltools` compatible object or `NULL`, if package is missing

**See Also**

Other `summary_functions`: [prep\\_combine\\_report\\_summaries\(\)](#), [prep\\_extract\\_classes\\_by\\_functions\(\)](#), [prep\\_extract\\_summary\(\)](#), [prep\\_extract\\_summary.dataquieR\\_result\(\)](#), [prep\\_extract\\_summary.dataquieR\\_result.dataquieR\\_result\(\)](#), [prep\\_render\\_pie\\_chart\\_from\\_summaryclasses\\_plotly\(\)](#), [prep\\_summary\\_to\\_classes\(\)](#)

---

`prep_render_pie_chart_from_summaryclasses_plotly`  
*Create a plotly pie chart*

---

**Description**

Create a plotly pie chart

**Usage**

```
prep_render_pie_chart_from_summaryclasses_plotly(  
  data,  
  meta_data = "item_level"  
)
```

**Arguments**

<code>data</code>	data as returned by <code>prep_summary_to_classes</code> but summarized by one column (currently, we support <code>indicator_metric</code> , <code>call_names</code> , <code>STUDY_SEGMENT</code> , and <code>VAR_NAMES</code> )
<code>meta_data</code>	<a href="#">meta_data</a>

**Value**

a `htmltools` compatible object

**See Also**

Other `summary_functions`: [prep\\_combine\\_report\\_summaries\(\)](#), [prep\\_extract\\_classes\\_by\\_functions\(\)](#), [prep\\_extract\\_summary\(\)](#), [prep\\_extract\\_summary.dataquieR\\_result\(\)](#), [prep\\_extract\\_summary.dataquieR\\_result.dataquieR\\_result\(\)](#), [prep\\_render\\_pie\\_chart\\_from\\_summaryclasses\\_ggplot2\(\)](#), [prep\\_summary\\_to\\_classes\(\)](#)

---

prep\_robust\_guess\_data\_type  
*Guess the data type of a vector*

---

**Description**

Guess the data type of a vector

**Usage**

```
prep_robust_guess_data_type(x, k = 50, it = 200)
```

**Arguments**

**x** a vector with characters  
**k** [numeric](#) sample size, if less than  $\text{floor}(\text{length}(x) / (\text{it}/20))$ , minimum sample size is 1.  
**it** [integer](#) number of iterations when taking samples

**Value**

a guess of the data type of  $x$ . An attribute `orig_type` is also attached to give the more detailed guess returned by [readr::guess\\_parser\(\)](#).

**Algorithm**

This function takes  $x$  and tries to guess the data type of random subsets of this vector using [readr::guess\\_parser\(\)](#). The RNG is initialized with a constant, so the function stays deterministic. It does such sub-sample based checks  $\text{it}$  times, the majority of the detected datatype determines the guessed data type.

**See Also**

[prep\\_datatype\\_from\\_data\(\)](#)

---

prep\_save\_report      *Save a dq\_report2*

---

**Description**

Save a dq\_report2

**Usage**

```
prep_save_report(report, file, compression_level = 3)
```

**Arguments**

report [dataquieR\\_resultset2](#) the report  
 file [character](#) the file name to write to  
 compression\_level [integer](#) from=0 to=9. Compression level. 9 is very slow.

**Value**

invisible(NULL)

---

prep\_scalelevel\_from\_data\_and\_metadata

*Heuristics to amend a SCALE\_LEVEL column and a UNIT column in the metadata*

---

**Description**

...if missing

**Usage**

```
prep_scalelevel_from_data_and_metadata(
  resp_vars = lifecycle::deprecated(),
  study_data,
  item_level = "item_level",
  label_col = LABEL,
  meta_data = item_level,
  meta_data_v2
)
```

**Arguments**

resp\_vars [variable list](#) deprecated, the function always addresses all variables.  
 study\_data [data.frame](#) the data frame that contains the measurements  
 item\_level [data.frame](#) the data frame that contains metadata attributes of study data  
 label\_col [variable attribute](#) the name of the column in the metadata with labels of variables  
 meta\_data [data.frame](#) old name for item\_level  
 meta\_data\_v2 [character](#) path to workbook like metadata file, see [prep\\_load\\_workbook\\_like\\_file](#) for details. **ALL LOADED DATAFRAMES WILL BE PURGED**, using [prep\\_purge\\_data\\_frame\\_cache](#), if you specify meta\_data\_v2.

**Value**

[data.frame](#) modified metadata

**Examples**

```
## Not run:
  prep_load_workbook_like_file("meta_data_v2")
  prep_scalelevel_from_data_and_metadata(study_data = "study_data")

## End(Not run)
```

---

```
prep_set_backend      Change the back-end of a report
```

---

**Description**

with this function, you can move a report from/to a storr storage.

**Usage**

```
prep_set_backend(r, storr_factory = NULL, amend = FALSE)
```

**Arguments**

`r` [dataquieR\\_resultset2](#) the report  
`storr_factory` storr the storr storage or NULL, to move the report fully back into the RAM.  
`amend` [logical](#) if there is already data in.storr\_factory, use it anyways – unsupported, so far!

**Value**

[dataquieR\\_resultset2](#) but now with the desired back-end

---

```
prep_study2meta      Guess a metadata data frame from study data.
```

---

**Description**

Guess a minimum metadata data frame from study data. Minimum required variable attributes are:

**Usage**

```
prep_study2meta(
  study_data,
  level = c(VARATT_REQUIRE_LEVELS$REQUIRED, VARATT_REQUIRE_LEVELS$RECOMMENDED),
  cumulative = TRUE,
  convert_factors = FALSE,
  guess_missing_codes = getOption("dataquieR.guess_missing_codes",
    dataquieR.guess_missing_codes_default),
  guess_character = getOption("dataquieR.guess_character", default =
    dataquieR.guess_character_default)
)
```

**Arguments**

**study\_data** [data.frame](#) the data frame that contains the measurements  
**level** [enum](#) levels to provide (see also [VARATT\\_REQUIRE\\_LEVELS](#))  
**cumulative** [logical](#) include attributes of all levels up to level  
**convert\_factors** [logical](#) convert factor columns to coded integers. if selected, then also the study data will be updated and returned.  
**guess\_missing\_codes** [logical](#) try to guess missing codes from the data  
**guess\_character** [logical](#) guess a data type for character columns based on the values

**Details**

```

dataquieR:::util_get_var_att_names_of_level(VARATT_REQUIRE_LEVELS$REQUIRED)
#>      VAR_NAMES      DATA_TYPE  MISSING_LIST_TABLE
#>      "VAR_NAMES"      "DATA_TYPE" "MISSING_LIST_TABLE"
  
```

The function also tries to detect missing codes.

**Value**

a `meta_data` data frame or a list with study data and metadata, if `convert_factors == TRUE`.

**Examples**

```

## Not run:
dataquieR::prep_study2meta(Orange, convert_factors = FALSE)

## End(Not run)
  
```

---

```
prep_summary_to_classes
```

*Classify metrics from a report summary table*

---

**Description**

Classify metrics from a report summary table

**Usage**

```
prep_summary_to_classes(report_summary)
```

**Arguments**

`report_summary` [list\(\)](#) as returned by [prep\\_extract\\_summary\(\)](#)

**Value**

`data.frame` classes for the report summary table, long format

**See Also**

Other summary\_functions: `prep_combine_report_summaries()`, `prep_extract_classes_by_functions()`, `prep_extract_summary()`, `prep_extract_summary.dataquieR_result()`, `prep_extract_summary.dataquieR_result.dataquieR_result()`, `prep_render_pie_chart_from_summaryclasses_ggplot2()`, `prep_render_pie_chart_from_summaryclasses_plotly()`

---

prep\_title\_escape      *Prepare a label as part of a title text for RMD files*

---

**Description**

Prepare a label as part of a title text for RMD files

**Usage**

```
prep_title_escape(s, html = FALSE)
```

**Arguments**

`s`                      the label  
`html`                    prepare the label for direct HTML output instead of RMD

**Value**

the escaped label

---

prep\_undisclose      *Remove data disclosing details*

---

**Description**

new function: no warranty, so far.

**Usage**

```
prep_undisclose(x, cores)
```

**Arguments**

`x`                      an object to un-disclose, a  
`cores`                    can be an integer with a number of cores to use. if not specified, the function uses the default cluster, if available and falls back to serial un-disclosing, otherwise.

**Value**

undisclosed object

---

prep\_unsplit\_val\_tabs *Combine all missing and value lists to one big table*

---

**Description**

Combine all missing and value lists to one big table

**Usage**

```
prep_unsplit_val_tabs(meta_data = "item_level", val_tab = NULL)
```

**Arguments**

meta\_data [data.frame](#) item level meta data to be used, defaults to "item\_level"  
 val\_tab [character](#) name of the table being created: This table will be added to the data frame cache (or overwritten). If NULL, the table will only be returned

**Value**

[data.frame](#) the combined table

---

prep\_valuelabels\_from\_data  
*Get value labels from data*

---

**Description**

Detects factors and converts them to compatible metadata/study data.

**Usage**

```
prep_valuelabels_from_data(resp_vars = colnames(study_data), study_data)
```

**Arguments**

resp\_vars [variable](#) names of the variables to fetch the value labels from the data  
 study\_data [data.frame](#) the data frame that contains the measurements

**Value**

a [list](#) with:

- VALUE\_LABELS: vector of value labels and modified study data
- ModifiedStudyData: study data with factors as integers

**Examples**

```
## Not run:  
dataquieR::prep_datatype_from_data(iris)  
  
## End(Not run)
```

---

```
print.dataquieR_result
```

*Print a [dataquieR](#) result returned by [dq\\_report2](#)*

---

**Description**

Print a [dataquieR](#) result returned by [dq\\_report2](#)

**Usage**

```
## S3 method for class 'dataquieR_result'  
print(x, ...)
```

**Arguments**

x	<a href="#">list</a> a <a href="#">dataquieR</a> result from <a href="#">dq_report2</a> or <a href="#">util_eval_to_dataquieR_result</a>
...	passed to <code>print</code> . Additionally, the argument slot may be passed to <code>print</code> only specific sub-results.

**Value**

see `print`

**See Also**

```
util_pretty_print()
```

---

```
print.dataquieR_resultset
```

*Generate a RMarkdown-based report from a [dataquieR](#) report*

---

**Description**

Generate a RMarkdown-based report from a [dataquieR](#) report

**Usage**

```
## S3 method for class 'dataquieR_resultset'  
print(...)
```

**Arguments**

... deprecated

**Value**

deprecated

---

```
print.dataquieR_resultset2
```

*Generate a HTML-based report from a [dataquieR](#) report*

---

**Description**

Generate a HTML-based report from a [dataquieR](#) report

**Usage**

```
## S3 method for class 'dataquieR_resultset2'
print(
  x,
  dir,
  view = TRUE,
  disable_plotly = FALSE,
  block_load_factor = getOption("dataquieR.print_block_load_factor",
    dataquieR.print_block_load_factor_default),
  advanced_options = list(),
  dashboard = NA,
  ...,
  cores = list(mode = "socket", logging = FALSE, cpus = util_detect_cores()),
  load_balancing = TRUE)
)
```

**Arguments**

x [dataquieR](#) report v2.

dir [character](#) directory to store the rendered report's files, a temporary one, if omitted. Directory will be created, if missing, files may be overwritten inside that directory

view [logical](#) display the report

disable\_plotly [logical](#) do not use plotly, even if installed

block\_load\_factor [numeric](#) see [dataquieR.print\\_block\\_load\\_factor](#)

advanced\_options [list](#) options to set during report computation, see [options\(\)](#)

dashboard      **logical** dashboard mode: TRUE: create a dashboard only, FALSE: don't create a dashboard at all, NA or missing: create a "normal" report with a dashboard included.

...              additional arguments:

cores            **integer** number of cpu cores to use or a named list with arguments for `parallelMap::parallelStart` or NULL, if parallel has already been started by the caller. Can also be a cluster.

**Value**

file names of the generated report's HTML files

---

```
print.dataquieR_summary
      Print a dataquieR summary
```

---

**Description**

Print a dataquieR summary

**Usage**

```
## S3 method for class 'dataquieR_summary'
print(
  x,
  ...,
  grouped_by = c("call_names", "indicator_metric"),
  dont_print = FALSE,
  folder_of_report = NULL,
  vars_to_include = c("study")
)
```

**Arguments**

x                the dataquieR summary, see `summary()` and `dq_report2()`

...              not yet used

grouped\_by      define the columns of the resulting matrix. It can be either "call\_names", one column per function, or "indicator\_metric", one column per indicator or both c("call\_names", "indicator\_metric"). The last combination is the default

dont\_print      suppress the actual printing, just return a printable object derived from x

folder\_of\_report      a named vector with the location of variable and call\_names

vars\_to\_include      "study", "ssi" or c("study", "ssi"). variables to include

**Value**

invisible html object

---

print.DataSlot      *Print a DataSlot object*

---

**Description**

Print a DataSlot object

**Usage**

```
## S3 method for class 'DataSlot'
print(x, ...)
```

**Arguments**

x	the object
...	not used

**Value**

see print

---

print.interval      *print implementation for the class interval*

---

**Description**

such objects, for now, only occur in RECCap rules, so this function is meant for internal use, mostly – for now.

**Usage**

```
## S3 method for class 'interval'
print(x, ...)
```

**Arguments**

x	interval objects to print
...	not used yet

**Value**

the printed object

**See Also**

base::print

---

print.list	<i>print a list of dataquieR_result objects</i>
------------	---

---

**Description**

print a list of dataquieR\_result objects

**Usage**

```
## S3 method for class 'list'  
print(x, ...)
```

**Arguments**

x	<a href="#">list()</a> only, if all elements inherit from <a href="#">dataquieR_result</a> , this implementation runs
...	passed to other implementations

**Value**

undefined

---

print.master_result	<i>Print a master_result object</i>
---------------------	-------------------------------------

---

**Description**

Print a master\_result object

**Usage**

```
## S3 method for class 'master_result'  
print(x, template = "default", ...)
```

**Arguments**

x	the object
template	the template for the iframes, not used, so far.
...	not used

**Value**

invisible(NULL)

---

```
print.numeric_with_unit
    Print a number with unit
```

---

**Description**

Print a number with unit

**Usage**

```
## S3 method for class 'numeric_with_unit'
print(x, ...)
```

**Arguments**

x	number with unit
...	not used

**Value**

```
invisible(x)
```

---

```
print.ReportSummaryTable
    print implementation for the class ReportSummaryTable
```

---

**Description**

Use this function to print results objects of the class ReportSummaryTable.

**Usage**

```
## S3 method for class 'ReportSummaryTable'
print(
  x,
  relative = lifecycle::deprecated(),
  dt = FALSE,
  fillContainer = FALSE,
  displayValues = FALSE,
  view = TRUE,
  drop = getOption("dataquieR.droplevels_ReportSummaryTable",
    dataquieR.droplevels_ReportSummaryTable_default),
  ...,
  flip_mode = "auto"
)
```

**Arguments**

x	ReportSummaryTable objects to print
relative	deprecated
dt	<a href="#">logical</a> use DT::datatables, if installed
fillContainer	<a href="#">logical</a> if dt is TRUE, control table size, see DT::datatables.
displayValues	<a href="#">logical</a> if dt is TRUE, also display the actual values
view	<a href="#">logical</a> if view is FALSE, do not print but return the output, only
drop	<a href="#">logical</a> if drop is FALSE, keep unused levels, see <a href="#">dataquieR.droplevels_ReportSummaryTable</a>
...	not used, yet
flip_mode	<a href="#">enum</a> default   flip   noflip   auto. Should the plot be in default orientation, flipped, not flipped or auto-flipped. Not all options are always supported. In general, this can be controlled by setting the <code>roptions(dataquieR.flip_mode = ...)</code> . If called from <code>dq_report</code> , you can also pass <code>flip_mode</code> to all function calls or set them specifically using <code>specific_args</code> .

**Value**

the printed object

**See Also**

`base::print`

---

print.Slot

*Print a Slot object*

---

**Description**

displays all warnings and stuff. then it prints x.

**Usage**

```
## S3 method for class 'Slot'
print(x, ...)
```

**Arguments**

x	the object
...	not used

**Value**

calls the next print method

print.StudyDataSlot    *Print a StudyDataSlot object*

---

**Description**

Print a StudyDataSlot object

**Usage**

```
## S3 method for class 'StudyDataSlot'  
print(x, ...)
```

**Arguments**

x	the object
...	not used

**Value**

see print

---

print.TableSlot    *Print a TableSlot object*

---

**Description**

Print a TableSlot object

**Usage**

```
## S3 method for class 'TableSlot'  
print(x, ...)
```

**Arguments**

x	the object
...	not used

**Value**

see print

---

```
print.util_pairs_ggplot_panels
      Print method for util_pairs_ggplot_panels objects
```

---

**Description**

Print method for util\_pairs\_ggplot\_panels objects

**Usage**

```
## S3 method for class 'util_pairs_ggplot_panels'
print(x, ...)
```

**Arguments**

x                    An object of class util\_pairs\_ggplot\_panels.  
...                   Ignored.

**Value**

The input object, invisibly.

---

```
progress_init_fkt        function to call on progress initialization
```

---

**Description**

has one argument, n, reporting the number of steps in the current job. needed, e.g., by packages, such as progressr. TODO

**See Also**

Other options: [dataquieR](#), [dataquieR.CONDITIONS\\_LEVEL\\_TRHESHOLD](#), [dataquieR.CONDITIONS\\_WITH\\_STACKTRACE](#), [dataquieR.ELEMENT\\_MISMATCH\\_CHECKTYPE](#), [dataquieR.ERRORS\\_WITH\\_CALLER](#), [dataquieR.GAM\\_for\\_LOESS](#), [dataquieR.MAHALANOBIS\\_THRESHOLD](#), [dataquieR.MAX\\_LABEL\\_LEN](#), [dataquieR.MAX\\_LONG\\_LABEL\\_LEN](#), [dataquieR.MAX\\_VALUE\\_LABEL\\_LEN](#), [dataquieR.MESSAGES\\_WITH\\_CALLER](#), [dataquieR.MULTIVARIATE\\_OUTLIER\\_CHECK](#), [dataquieR.VALUE\\_LABELS\\_htmlescaped](#), [dataquieR.WARNINGS\\_WITH\\_CALLER](#), [dataquieR.acc\\_loess.exclude\\_const](#), [dataquieR.acc\\_loess.mark\\_time\\_points](#), [dataquieR.acc\\_loess.min\\_bw](#), [dataquieR.acc\\_loess.min\\_proportion](#), [dataquieR.acc\\_loess.plot\\_format](#), [dataquieR.acc\\_loess.plot\\_observations](#), [dataquieR.acc\\_margins\\_num](#), [dataquieR.acc\\_margins\\_sort](#), [dataquieR.acc\\_multivariate\\_outlier.scale](#), [dataquieR.col\\_con\\_con\\_empirical](#), [dataquieR.col\\_con\\_con\\_logical](#), [dataquieR.convert\\_to\\_list\\_for\\_lapply](#), [dataquieR.debug](#), [dataquieR.des\\_summary\\_hard\\_lim\\_remove](#), [dataquieR.dontwrapresults](#), [dataquieR.droplevels\\_ReportSummaryT](#), [dataquieR.dt\\_adjust](#), [dataquieR.fix\\_column\\_type\\_on\\_read](#), [dataquieR.flip\\_mode](#), [dataquieR.force\\_item\\_spec](#), [dataquieR.force\\_label\\_col](#), [dataquieR.grading\\_formats](#), [dataquieR.grading\\_rulesets](#), [dataquieR.guess\\_character](#), [dataquieR.guess\\_missing\\_codes](#), [dataquieR.ignore\\_empty\\_vars](#),

```

dataquieR.lang, dataquieR.lazy_plots, dataquieR.lazy_plots_cache, dataquieR.lazy_plots_gg_compatibility,
dataquieR.locale, dataquieR.max_cat_resp_var_levels_in_plot, dataquieR.max_group_var_levels_in_plot,
dataquieR.max_group_var_levels_with_violins, dataquieR.min_obs_per_group_var_in_plot,
dataquieR.min_time_points_for_cat_resp_var, dataquieR.non_disclosure, dataquieR.old_factor_handling,
dataquieR.old_type_adjust, dataquieR.precomputeStudyData, dataquieR.print_block_load_factor,
dataquieR.progress_fkt_default, dataquieR.progress_msg_fkt_default, dataquieR.resume_checkpoint,
dataquieR.resume_print, dataquieR.scale_level_heuristics_control_binaryrecodelimit,
dataquieR.scale_level_heuristics_control_metriclevels, dataquieR.study_data_cache_max,
dataquieR.study_data_cache_metrics, dataquieR.study_data_cache_metrics_env, dataquieR.study_data_cache,
dataquieR.study_data_colnames_case_sensitive, dataquieR.testdebug, dataquieR.traceback,
dataquieR.type_adjust_parallel

```

---

```
pro_applicability_matrix
```

*Check applicability of DQ functions on study data*

---

### Description

Checks applicability of DQ functions based on study data and metadata characteristics

### Usage

```

pro_applicability_matrix(
  study_data,
  item_level = "item_level",
  split_segments = FALSE,
  label_col,
  max_vars_per_plot = 20,
  meta_data_segment,
  meta_data_dataframe,
  flip_mode = "noflip",
  meta_data_v2,
  meta_data = item_level,
  segment_level,
  dataframe_level
)

```

### Arguments

`study_data` [data.frame](#) the data frame that contains the measurements

`item_level` [data.frame](#) the data frame that contains metadata attributes of study data

`split_segments` [logical](#) return one matrix per study segment

`label_col` [variable attribute](#) the name of the column in the metadata with labels of variables

`max_vars_per_plot` [integer](#) from=0. The maximum number of variables per single plot.

meta_data_segment	<a href="#">data.frame</a> – optional: Segment level metadata
meta_data_dataframe	<a href="#">data.frame</a> – optional: Data frame level metadata
flip_mode	<a href="#">enum</a> default   flip   noflip   auto. Should the plot be in default orientation, flipped, not flipped or auto-flipped. Not all options are always supported. In general, this can be controlled by setting the <code>roptions(dataquieR.flip_mode = ...)</code> . If called from <code>dq_report</code> , you can also pass <code>flip_mode</code> to all function calls or set them specifically using <code>specific_args</code> .
meta_data_v2	<a href="#">character</a> path to workbook like metadata file, see <a href="#">prep_load_workbook_like_file</a> for details. <b>ALL LOADED DATAFRAMES WILL BE PURGED</b> , using <a href="#">prep_purge_data_frame_cache</a> , if you specify <code>meta_data_v2</code> .
meta_data	<a href="#">data.frame</a> old name for <code>item_level</code>
segment_level	<a href="#">data.frame</a> alias for <code>meta_data_segment</code>
dataframe_level	<a href="#">data.frame</a> alias for <code>meta_data_dataframe</code>

## Details

This is a preparatory support function that compares study data with associated metadata. A prerequisite of this function is that the no. of columns in the study data complies with the no. of rows in the metadata.

For each existing R-implementation, the function searches for necessary static metadata and returns a heatmap like matrix indicating the applicability of each data quality implementation.

In addition, the data type defined in the metadata is compared with the observed data type in the study data.

## Value

a list with:

- `SummaryTable`: data frame about the applicability of each indicator function (each function in a column). its [integer](#) values can be one of the following four categories: 0. Non-matching datatype + Incomplete metadata, 1. Non-matching datatype + complete metadata, 2. Matching datatype + Incomplete metadata, 3. Matching datatype + complete metadata, 4. Not applicable according to data type
- `ApplicabilityPlot`: [ggplot2::ggplot2](#) heatmap plot, graphical representation of `SummaryTable`
- `ApplicabilityPlotList`: [list](#) of plots per (maybe artificial) segment
- `ReportSummaryTable`: data frame underlying `ApplicabilityPlot`

---

```

rbind.ReportSummaryTable
      Combine ReportSummaryTable outputs

```

---

**Description**

Using this rbind implementation, you can combine different heatmap-like results of the class ReportSummaryTable.

**Usage**

```

## S3 method for class 'ReportSummaryTable'
rbind(...)

```

**Arguments**

```

...          ReportSummaryTable objects to combine.

```

**See Also**

[base::rbind.data.frame](#)

---

```

RELCOMPL_SPEED      Cross-item level metadata attribute name

```

---

**Description**

```

TODO
TODO

```

**See Also**

[meta\\_data\\_cross](#)

[meta\\_data\\_cross](#)

Other meta\_data\_cross: ASSOCIATION\_DIRECTION, ASSOCIATION\_FORM, ASSOCIATION\_METRIC, ASSOCIATION\_RANGE, CHECK\_ID, CHECK\_LABEL, COMPUTED\_VARIABLE\_ROLES, CONTRADICTION\_TERM, CONTRADICTION\_TYPE, DATA\_PREPARATION, GOLDSTANDARD, IRV, MAHALANOBIS\_THRESHOLD, MAXIMUM\_LONG\_STRING, MISS\_RESP, MULTIVARIATE\_OUTLIER\_CHECK, MULTIVARIATE\_OUTLIER\_CHECKTYPE, REL\_VAL, RESPT\_PER\_ITEM, SCALE\_ACRONYM, SCALE\_NAME, TOTRESPT, VARIABLE\_LIST, VARIABLE\_LIST\_ORDER, meta\_data\_computation, meta\_data\_cross

Other SSI: COMPUTED\_VARIABLE\_ROLES, IRV, MAXIMUM\_LONG\_STRING, MISS\_RESP, RESPT\_PER\_ITEM, TOTRESPT

Other meta\_data\_cross: ASSOCIATION\_DIRECTION, ASSOCIATION\_FORM, ASSOCIATION\_METRIC, ASSOCIATION\_RANGE, CHECK\_ID, CHECK\_LABEL, COMPUTED\_VARIABLE\_ROLES, CONTRADICTION\_TERM, CONTRADICTION\_TYPE,

DATA\_PREPARATION, GOLDSTANDARD, IRV, MAHALANOBIS\_THRESHOLD, MAXIMUM\_LONG\_STRING, MISS\_RESP, MULTIVARIATE\_OUTLIER\_CHECK, MULTIVARIATE\_OUTLIER\_CHECKTYPE, REL\_VAL, RESPT\_PER\_ITEM, SCALE\_ACRONYM, SCALE\_NAME, TOTRESPT, VARIABLE\_LIST, VARIABLE\_LIST\_ORDER, meta\_data\_computation, meta\_data\_cross

Other SSI: COMPUTED\_VARIABLE\_ROLES, IRV, MAXIMUM\_LONG\_STRING, MISS\_RESP, RESPT\_PER\_ITEM, TOTRESPT

---

REL\_VAL

*Cross-item level metadata attribute name*

---

### Description

Specifies the type of reliability or validity analysis. The string specifies the analysis algorithm to be used, and can be either "inter-class" or "intra-class".

### Usage

REL\_VAL

### Format

An object of class character of length 1.

### See Also

[meta\\_data\\_cross](#)

Other meta\_data\_cross: ASSOCIATION\_DIRECTION, ASSOCIATION\_FORM, ASSOCIATION\_METRIC, ASSOCIATION\_RANGE, CHECK\_ID, CHECK\_LABEL, COMPUTED\_VARIABLE\_ROLES, CONTRADICTION\_TERM, CONTRADICTION\_TYPE, DATA\_PREPARATION, GOLDSTANDARD, IRV, MAHALANOBIS\_THRESHOLD, MAXIMUM\_LONG\_STRING, MISS\_RESP, MULTIVARIATE\_OUTLIER\_CHECK, MULTIVARIATE\_OUTLIER\_CHECKTYPE, RELCOMPL\_SPEED, RESPT\_PER\_ITEM, SCALE\_ACRONYM, SCALE\_NAME, TOTRESPT, VARIABLE\_LIST, VARIABLE\_LIST\_ORDER, meta\_data\_computation, meta\_data\_cross

---

resnames

*Return names of result slots (e.g., 3rd dimension of dataquieR results)*

---

### Description

Return names of result slots (e.g., 3rd dimension of dataquieR results)

### Usage

resnames(x)

**Arguments**

x                    the objects

**Value**

character vector with names

---

```
resnames.dataquieR_resultset2
```

*Return names of result slots (e.g., 3rd dimension of dataquieR results)*

---

**Description**

Return names of result slots (e.g., 3rd dimension of dataquieR results)

**Usage**

```
## S3 method for class 'dataquieR_resultset2'
resnames(x)
```

**Arguments**

x                    the objects

**Value**

character vector with names

---

```
RESPT_PER_ITEM
```

*Cross-item level metadata attribute name*

---

**Description**

TODO

**See Also**

[meta\\_data\\_cross](#)

Other [meta\\_data\\_cross](#): [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

Other SSI: [COMPUTED\\_VARIABLE\\_ROLES](#), [IRV](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [RELCOMPL\\_SPEED](#), [TOTRESPT](#)

---

SCALE_ACRONYM	<i>Cross-item level metadata attribute name TODO</i>
---------------	--

---

**Description**

Cross-item level metadata attribute name TODO

**Usage**

SCALE\_ACRONYM

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_cross](#)

Other [meta\\_data\\_cross](#): [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

---

SCALE_LEVELS	<i>Scale Levels</i>
--------------	---------------------

---

**Description**

**Scale Levels of Study Data according to Stevens' s Typology:**

In the metadata, the following entries are allowed for the [variable attribute SCALE\\_LEVEL](#):

**Usage**

SCALE\_LEVELS

**Format**

An object of class list of length 5.

**Details**

- nominal for categorical variables
- ordinal for ordinal variables (i.e., comparison of values is possible)
- interval for interval scales, i.e., distances are meaningful
- ratio for ratio scales, i.e., ratios are meaningful
- na for variables, that contain e.g. unstructured texts, json, xml, ... to distinguish them from variables, that still need to have the SCALE\_LEVEL estimated by `prep_scalelevel_from_data_and_metadata()`

**Examples:**

- sex, eye color – nominal
- income group, education level – ordinal
- temperature in degree Celsius – interval
- body weight, temperature in Kelvin – ratio

**See Also**

[Wikipedia](#)

---

SCALE\_NAME

*Cross-item level metadata attribute name TODO*

---

**Description**

Cross-item level metadata attribute name TODO

**Usage**

SCALE\_NAME

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_cross](#)

Other `meta_data_cross`: [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

---

SEGMENT\_ID\_REF\_TABLE    *Segment level metadata attribute name*

---

**Description**

The name of the data frame containing the reference IDs to be compared with the IDs in the targeted segment.

**Usage**

SEGMENT\_ID\_REF\_TABLE

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_segment](#)

---

SEGMENT\_ID\_TABLE    *Deprecated segment level metadata attribute name*

---

**Description**

The name of the data frame containing the reference IDs to be compared with the IDs in the targeted segment.

**Usage**

SEGMENT\_ID\_TABLE

**Format**

An object of class character of length 1.

**Details**

Please use [SEGMENT\\_ID\\_REF\\_TABLE](#)

---

SEGMENT_ID_VARS	<i>Segment level metadata attribute name</i>
-----------------	--

---

**Description**

All variables that are to be used as one single ID variable (combined key) in a segment.

**Usage**

SEGMENT\_ID\_VARS

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_segment](#)

---

SEGMENT_MISS	<i>Segment level metadata attribute name</i>
--------------	--

---

**Description**

true or false to suppress crude segment missingness output (Completeness/Misg. Segments in the report). Defaults to compute the output, if more than one segment is available in the item-level metadata.

**Usage**

SEGMENT\_MISS

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_segment](#)

---

SEGMENT_PART_VARS	<i>Segment level metadata attribute name</i>
-------------------	--

---

**Description**

The name of the segment participation status variable

**Usage**

SEGMENT\_PART\_VARS

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_segment](#)

---

SEGMENT_RECORD_CHECK	<i>Segment level metadata attribute name</i>
----------------------	--

---

**Description**

The type of check to be conducted when comparing the reference ID table with the IDs in a segment.

**Usage**

SEGMENT\_RECORD\_CHECK

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_segment](#)

---

SEGMENT\_RECORD\_COUNT    *Segment level metadata attribute name*

---

**Description**

Number of expected data records in each segment. [numeric](#). Check only conducted if number entered

**Usage**

SEGMENT\_RECORD\_COUNT

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_segment](#)

---

SEGMENT\_UNIQUE\_ID    *Segment level metadata attribute name*

---

**Description**

Segment level metadata attribute name

**Usage**

SEGMENT\_UNIQUE\_ID

**Format**

An object of class character of length 1.

**See Also**

[DF\\_UNIQUE\\_ID](#)  
[meta\\_data\\_segment](#)

---

SEGMENT_UNIQUE_ROWS	<i>Segment level metadata attribute name</i>
---------------------	--

---

**Description**

Specifies whether identical data is permitted across rows in a segment (excluding ID variables)

**Usage**

SEGMENT\_UNIQUE\_ROWS

**Format**

An object of class character of length 1.

**See Also**

[meta\\_data\\_segment](#)

---

SPLIT_CHAR	<i>Character used by default as a separator in metadata such as missing codes</i>
------------	---

---

**Description**

This 1 character is according to our metadata concept "I".

**Usage**

SPLIT\_CHAR

**Format**

An object of class character of length 1.

---

study_data	<i>Data frame with the study data whose quality is being assessed</i>
------------	---

---

**Description**

Study data is expected in wide format. It should contain all variables for all segments in one large table, even, if some variables are not measured for all observational units (study participants).

---

```
summary.dataquieR_resultset
      Summarize a dataquieR report
```

---

**Description**

Deprecated

**Usage**

```
## S3 method for class 'dataquieR_resultset'
summary(...)
```

**Arguments**

...                   Deprecated

**Value**

Deprecated

---

```
summary.dataquieR_resultset2
      Generate a report summary table
```

---

**Description**

Generate a report summary table

**Usage**

```
## S3 method for class 'dataquieR_resultset2'
summary(
  object,
  aspect = c("applicability", "error", "anamat", "indicator_or_descriptor"),
  FUN,
  collapse = "\n<br />\n",
  ...
)
```

**Arguments**

object	a square result set
aspect	an aspect/problem category of results
FUN	function to apply to the cells of the result table
collapse	passed to FUN
...	not used

**Value**

a summary of a dataquieR report

**Examples**

```
## Not run:
  util_html_table(summary(report),
    filter = "top", options = list(scrollCollapse = TRUE, scrolly = "75vh"),
    is_matrix_table = TRUE, rotate_headers = TRUE, output_format = "HTML"
  )

## End(Not run)
```

---

 TOTRESPT

---

*Cross-item level metadata attribute name*


---

**Description**

TODO

**See Also**

[meta\\_data\\_cross](#)

Other [meta\\_data\\_cross](#): [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [VARIABLE\\_LIST](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

Other SSI: [COMPUTED\\_VARIABLE\\_ROLES](#), [IRV](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [RELCOMPL\\_SPEED](#), [RESPT\\_PER\\_ITEM](#)

---

 UNITS

---

*Valid unit symbols according to `units::valid_udunits()`*


---

**Description**

like m, g, N, ...

**See Also**

Other UNITS: [UNIT\\_IS\\_COUNT](#), [UNIT\\_PREFIXES](#), [UNIT\\_PREFIX\\_FACTORS](#), [UNIT\\_SOURCES](#), [WELL\\_KNOWN\\_META\\_VARIABLE\\_NA](#)

---

UNIT\_IS\_COUNT      *Is a unit a count according to* `units::valid_udunits()`

---

**Description**

see column def, therein

**Details**

like %, ppt, ppm

**See Also**

Other UNITS: [UNITS](#), [UNIT\\_PREFIXES](#), [UNIT\\_PREFIX\\_FACTORS](#), [UNIT\\_SOURCES](#), [WELL\\_KNOWN\\_META\\_VARIABLE\\_NAMES](#)

---

UNIT\_PREFIXES      *Valid      unit      prefixes      according      to*  
[units::valid\\_udunits\\_prefixes\(\)](#)

---

**Description**

like k, m, M, c, ...

**See Also**

Other UNITS: [UNITS](#), [UNIT\\_IS\\_COUNT](#), [UNIT\\_PREFIX\\_FACTORS](#), [UNIT\\_SOURCES](#), [WELL\\_KNOWN\\_META\\_VARIABLE\\_NAMES](#)

---

UNIT\_PREFIX\_FACTORS      *Factors related to unit prefixes* `units::valid_udunits_prefixes()`

---

**Description**

named numeric vector

**Details**

translates k, m, M, c, ... to 1000, 0.001, ...

**See Also**

Other UNITS: [UNITS](#), [UNIT\\_IS\\_COUNT](#), [UNIT\\_PREFIXES](#), [UNIT\\_SOURCES](#), [WELL\\_KNOWN\\_META\\_VARIABLE\\_NAMES](#)

---

UNIT_SOURCES	<i>Maturity stage of a unit according to <code>units::valid_udunits()</code></i>
--------------	--

---

**Description**

see column `source_xml` therein, i.e., base, derived, accepted, or common

**See Also**

Other UNITS: [UNITS](#), [UNIT\\_IS\\_COUNT](#), [UNIT\\_PREFIXES](#), [UNIT\\_PREFIX\\_FACTORS](#), [WELL\\_KNOWN\\_META\\_VARIABLE\\_NAMES](#)

---

value/missing-lists	<i>Data frame with labels for missing- and jump-codes #' Metadata about value and missing codes</i>
---------------------	---

---

**Description**

`data.frame` with the following columns:

- `CODE_VALUE`: [numeric](#) | [DATETIME](#) Missing or categorical code (the number or date representing a missing/category)
- `CODE_LABEL`: [character](#) a label for the missing code or category
- `CODE_CLASS`: [enum](#) JUMP | MISSING. For missing lists: Class of the missing code.
- `CODE_INTERPRET` [enum](#) I | P | PL | R | BO | NC | O | UH | UO | NE. For missing lists: Class of the missing code according to [AAPOR](#).
- `resp_vars`: [character](#) For missing lists: optional, if a missing code is specific for some variables, it is listed for each such variable with one entry in `resp_vars`, If NA, the code is assumed shared among all variables. For v1.0 metadata, you need to refer to `VAR_NAMES` here.

**See Also**

[Online](#)

[com\\_item\\_missingness\(\)](#)

[com\\_segment\\_missingness\(\)](#)

[com\\_qualified\\_item\\_missingness\(\)](#)

[com\\_qualified\\_segment\\_missingness\(\)](#)

[con\\_inadmissible\\_categorical\(\)](#)

[con\\_inadmissible\\_vocabulary\(\)](#)

[MISSING\\_LIST\\_TABLE](#)

[VALUE\\_LABEL\\_TABLE](#)

[STANDARDIZED\\_VOCABULARY\\_TABLE](#)

[cause\\_label\\_df](#)

---

VARATT\_REQUIRE\_LEVELS *Requirement levels of certain metadata columns*

---

### Description

These levels are cumulatively used by the function [prep\\_create\\_meta](#) and related in the argument level therein.

### Usage

VARATT\_REQUIRE\_LEVELS

### Format

An object of class `list` of length 5.

### Details

currently available:

- 'COMPATIBILITY' = "compatibility"
- 'REQUIRED' = "required"
- 'RECOMMENDED' = "recommended"
- 'OPTIONAL' = "optional"
- 'TECHNICAL' = "technical"

---

VARIABLE\_LIST *Cross-item level metadata attribute name*

---

### Description

Specifies a group of variables for multivariate analyses. Separated by `|`, please use variable names from [VAR\\_NAMES](#) or a label as specified in `label_col`, usually [LABEL](#) or [LONG\\_LABEL](#).

### Usage

VARIABLE\_LIST

### Format

An object of class `character` of length 1.

### Details

if missing, `dataquieR` will create such IDs from [CONTRADICTION\\_TERM](#), if specified.

**See Also**[meta\\_data\\_cross](#)

Other [meta\\_data\\_cross](#): [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST\\_ORDER](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

---

VARIABLE\_LIST\_ORDER      *Cross-item level metadata attribute name TODO internal use, only*

---

**Description**

Cross-item level metadata attribute name TODO internal use, only

**Usage**

VARIABLE\_LIST\_ORDER

**Format**

An object of class character of length 1.

**See Also**[meta\\_data\\_cross](#)

Other [meta\\_data\\_cross](#): [ASSOCIATION\\_DIRECTION](#), [ASSOCIATION\\_FORM](#), [ASSOCIATION\\_METRIC](#), [ASSOCIATION\\_RANGE](#), [CHECK\\_ID](#), [CHECK\\_LABEL](#), [COMPUTED\\_VARIABLE\\_ROLES](#), [CONTRADICTION\\_TERM](#), [CONTRADICTION\\_TYPE](#), [DATA\\_PREPARATION](#), [GOLDSTANDARD](#), [IRV](#), [MAHALANOBIS\\_THRESHOLD](#), [MAXIMUM\\_LONG\\_STRING](#), [MISS\\_RESP](#), [MULTIVARIATE\\_OUTLIER\\_CHECK](#), [MULTIVARIATE\\_OUTLIER\\_CHECKTYPE](#), [RELCOMPL\\_SPEED](#), [REL\\_VAL](#), [RESPT\\_PER\\_ITEM](#), [SCALE\\_ACRONYM](#), [SCALE\\_NAME](#), [TOTRESPT](#), [VARIABLE\\_LIST](#), [meta\\_data\\_computation](#), [meta\\_data\\_cross](#)

---

VARIABLE\_ROLES      *Variable roles can be one of the following:*

---

**Description**

- intro a variable holding consent-data
- primary a primary outcome variable
- secondary a secondary outcome variable
- process a variable describing the measurement process
- suppress a variable added on the fly computing sub-reports, i.e., by [dq\\_report\\_by](#) to have all referred variables available, even if they are not part of the currently processed segment. But they will only be fully assessed in their real segment's report.

**Usage**

VARIABLE\_ROLES

**Format**An object of class `list` of length 5.

---

`WELL_KNOWN_META_VARIABLE_NAMES`*Well-known metadata column names, names of metadata columns*

---

**Description**

names of the variable attributes in the metadata frame holding the names of the respective observers, devices, lower limits for plausible values, upper limits for plausible values, lower limits for allowed values, upper limits for allowed values, the variable name (column name, e.g. v0020349) used in the study data, the variable name used for processing (readable name, e.g. RR\_DIAST\_1) and in parameters of the QA-Functions, the variable label, variable long label, variable short label, variable data type (see also [DATA\\_TYPES](#)), re-code for definition of lists of event categories, missing lists and jump lists as CSV strings. For valid units see [UNITS](#).

**Usage**`WELL_KNOWN_META_VARIABLE_NAMES`**Format**An object of class `list` of length 63.**Details**

all entries of this list will be mapped to the package's exported `NAMESPACE` environment directly, i.e. they are available directly by their names too:

- [VAR\\_NAMES](#)
- [LABEL](#)
- [DATA\\_TYPE](#)
- [SCALE\\_LEVEL](#)
- [UNIT](#)
- [VALUE\\_LABELS](#)
- [VALUE\\_LABEL\\_TABLE](#)
- [MISSING\\_LIST](#)
- [JUMP\\_LIST](#)
- [MISSING\\_LIST\\_TABLE](#)

- HARD\_LIMITS
- DETECTION\_LIMITS
- SOFT\_LIMITS
- CONTRADICTIONS
- DISTRIBUTION
- DECIMALS
- DATA\_ENTRY\_TYPE
- END\_DIGIT\_CHECK
- CO\_VARS
- GROUP\_VAR\_OBSERVER
- GROUP\_VAR\_DEVICE
- KEY\_OBSERVER
- KEY\_DEVICE
- TIME\_VAR
- TIME\_VAR\_END
- KEY\_DATETIME
- PART\_VAR
- STUDY\_SEGMENT
- KEY\_STUDY\_SEGMENT
- VARIABLE\_ROLE
- VARIABLE\_ORDER
- LONG\_LABEL
- SOFT\_LIMIT\_LOW
- SOFT\_LIMIT\_UP
- HARD\_LIMIT\_LOW
- HARD\_LIMIT\_UP
- DETECTION\_LIMIT\_LOW
- DETECTION\_LIMIT\_UP
- INCL\_SOFT\_LIMIT\_LOW
- INCL\_SOFT\_LIMIT\_UP
- INCL\_HARD\_LIMIT\_LOW
- INCL\_HARD\_LIMIT\_UP
- LOCATION\_RANGE
- LOCATION\_METRIC
- PROPORTION\_RANGE
- LOCATION\_LIMIT\_LOW
- LOCATION\_LIMIT\_UP

- INCL\_LOCATION\_LIMIT\_LOW
- INCL\_LOCATION\_LIMIT\_UP
- PROPORTION\_LIMIT\_LOW
- PROPORTION\_LIMIT\_UP
- INCL\_PROPORTION\_LIMIT\_LOW
- INCL\_PROPORTION\_LIMIT\_UP
- RECODE\_CASES
- RECODE\_CONTROL
- GRADING\_RULESET
- STANDARDIZED\_VOCABULARY\_TABLE
- DATAFRAMES
- ENCODING
- UNIVARIATE\_OUTLIER\_CHECKTYPE
- N\_RULES
- EXTENDED\_DATA\_TYPE
- COMPUTED\_VARIABLE\_ROLE

### See Also

[meta\\_data\\_segment](#) for STUDY\_SEGMENT

[online](#)

Other UNITS: [UNITS](#), [UNIT\\_IS\\_COUNT](#), [UNIT\\_PREFIXES](#), [UNIT\\_PREFIX\\_FACTORS](#), [UNIT\\_SOURCES](#)

### Examples

```
print(WELL_KNOWN_META_VARIABLE_NAMES$VAR_NAMES)
# print(VAR_NAMES) # should usually also work
```

---

```
[.dataquieR_resultset2
```

*Get a subset of a dataquieR dq\_report2 report*

---

### Description

Get a subset of a dataquieR dq\_report2 report

### Usage

```
## S3 method for class 'dataquieR_resultset2'
x[row, col, res, drop = FALSE, els = row, as_raw = FALSE]
```

**Arguments**

x	the report
row	the variable names, must be unique
col	the function-call-names, must be unique
res	the result slot, must be unique
drop	drop, if length is 1
els	used, if in list-mode with named argument
as_raw	retrieve the result maybe as compressed raw util_compress() serialized object

**Value**

a list with results, depending on drop and the number of results, the list may contain all requested results in sub-lists. The order of the results follows the order of the row/column/result-names given

---

```
[<-dataquieR_resultset2
      Write to a report
```

---

**Description**

Overwriting of elements only list-wise supported

**Usage**

```
## S3 replacement method for class 'dataquieR_resultset2'
x[...] <- value
```

**Arguments**

x	a 'dataquieR_resultset2'
...	if this contains only one entry and this entry is not named or its name is els, then, the report will be accessed in list mode.
value	new value to write

**Value**

nothing, stops

---

```
[[.dataquieR_resultset2
```

*Get a single result from a dataquieR 2 report*

---

### Description

Get a single result from a dataquieR 2 report

### Usage

```
## S3 method for class 'dataquieR_resultset2'
x[[e1]]
```

### Arguments

x	the report
e1	the index

### Value

the dataquieR result object

---

```
[[<-.dataquieR_resultset2
```

*Set a single result from a dataquieR 2 report*

---

### Description

Set a single result from a dataquieR 2 report

### Usage

```
## S3 replacement method for class 'dataquieR_resultset2'
x[[e1]] <- value
```

### Arguments

x	the report
e1	the index
value	the single result

### Value

the dataquieR result object

---

\$.dataquieR\_resultset2

*Access single results from a [dataquieR\\_resultset2](#) report*

---

### Description

Access single results from a [dataquieR\\_resultset2](#) report

### Usage

```
## S3 method for class 'dataquieR_resultset2'  
x$el
```

### Arguments

x	the report
el	the index

### Value

the dataquieR result object

---

\$<-.dataquieR\_resultset2

*Write single results from a [dataquieR\\_resultset2](#) report*

---

### Description

Write single results from a [dataquieR\\_resultset2](#) report

### Usage

```
## S3 replacement method for class 'dataquieR_resultset2'  
x$el <- value
```

### Arguments

x	the report
el	the index
value	the single result

### Value

the dataquieR result object

---

`%/.numeric_with_unit` *Operator caring for units*

---

**Description**

Operator caring for units

**Usage**

```
## S3 method for class 'numeric_with_unit'  
e1 %/% e2
```

**Arguments**

e1	first argument
e2	second argument

**Value**

result

---

`%/%.numeric_with_unit` *Operator caring for units*

---

**Description**

Operator caring for units

**Usage**

```
## S3 method for class 'numeric_with_unit'  
e1 %/% e2
```

**Arguments**

e1	first argument
e2	second argument

**Value**

result

---

`^.numeric_with_unit`    *Operator caring for units*

---

**Description**

Operator caring for units

**Usage**

```
## S3 method for class 'numeric_with_unit'  
e1 ^ e2
```

**Arguments**

<code>e1</code>	first argument
<code>e2</code>	second argument

**Value**

result

# Index

## \* SSI

COMPUTED\_VARIABLE\_ROLES, 46  
IRV, 165  
MAXIMUM\_LONG\_STRING, 166  
MISS\_RESP, 168  
RELCOMPL\_SPEED, 236  
RESPT\_PER\_ITEM, 238  
TOTRESPT, 247

## \* UNITS

UNIT\_IS\_COUNT, 248  
UNIT\_PREFIX\_FACTORS, 248  
UNIT\_PREFIXES, 248  
UNIT\_SOURCES, 249  
UNITS, 247  
WELL\_KNOWN\_META\_VARIABLE\_NAMES,  
252

## \* data-frame-cache

prep\_add\_data\_frames, 175  
prep\_get\_data\_frame, 198  
prep\_list\_dataframes, 203  
prep\_load\_folder\_with\_metadata,  
205  
prep\_load\_workbook\_like\_file, 207  
prep\_purge\_data\_frame\_cache, 215  
prep\_remove\_from\_cache, 216

## \* datasets

ASSOCIATION\_DIRECTION, 40  
ASSOCIATION\_FORM, 41  
ASSOCIATION\_METRIC, 41  
ASSOCIATION\_RANGE, 42  
CHECK\_ID, 42  
CHECK\_LABEL, 43  
CODE\_CLASSES, 44  
CODE\_LIST\_TABLE, 44  
CODE\_ORDER, 45  
COMPUTATION\_RULE, 45  
contradiction\_functions\_descriptions,  
54  
CONTRADICTION\_TERM, 55

CONTRADICTION\_TYPE, 55

DATA\_PREPARATION, 122

DATA\_TYPES, 123

DATA\_TYPES\_OF\_R\_TYPE, 124

dataquieR.applicability\_problem,  
72

dataquieR.intrinsic\_applicability\_problem,  
90

dataquieR.study\_data\_cache\_metrics\_env\_default,  
115

DF\_CODE, 130

DF\_ELEMENT\_COUNT, 130

DF\_ID\_REF\_TABLE, 131

DF\_ID\_VARS, 131

DF\_NAME, 132

DF\_RECORD\_CHECK, 132

DF\_RECORD\_COUNT, 133

DF\_UNIQUE\_ID, 133

DF\_UNIQUE\_ROWS, 134

dimensions, 135

dims, 136

DISTRIBUTIONS, 136

GOLDSTANDARD, 146

MAHALANOBIS\_THRESHOLD, 165

MULTIVARIATE\_OUTLIER\_CHECK, 168

MULTIVARIATE\_OUTLIER\_CHECKTYPE,  
169

REL\_VAL, 237

SCALE\_ACRONYM, 239

SCALE\_LEVELS, 239

SCALE\_NAME, 240

SEGMENT\_ID\_REF\_TABLE, 241

SEGMENT\_ID\_TABLE, 241

SEGMENT\_ID\_VARS, 242

SEGMENT\_MISS, 242

SEGMENT\_PART\_VARS, 243

SEGMENT\_RECORD\_CHECK, 243

SEGMENT\_RECORD\_COUNT, 244

SEGMENT\_UNIQUE\_ID, 244

- SEGMENT\_UNIQUE\_ROWS, 245
- SPLIT\_CHAR, 245
- VARATT\_REQUIRE\_LEVELS, 250
- VARIABLE\_LIST, 250
- VARIABLE\_LIST\_ORDER, 251
- VARIABLE\_ROLES, 251
- WELL\_KNOWN\_META\_VARIABLE\_NAMES, 252
- \* **meta\_data\_computation**
  - COMPUTATION\_RULE, 45
- \* **meta\_data\_cross**
  - ASSOCIATION\_DIRECTION, 40
  - ASSOCIATION\_FORM, 41
  - ASSOCIATION\_METRIC, 41
  - ASSOCIATION\_RANGE, 42
  - CHECK\_ID, 42
  - CHECK\_LABEL, 43
  - COMPUTED\_VARIABLE\_ROLES, 46
  - CONTRADICTION\_TERM, 55
  - CONTRADICTION\_TYPE, 55
  - DATA\_PREPARATION, 122
  - GOLDSTANDARD, 146
  - IRV, 165
  - MAHALANOBIS\_THRESHOLD, 165
  - MAXIMUM\_LONG\_STRING, 166
  - meta\_data\_computation, 167
  - meta\_data\_cross, 167
  - MISS\_RESP, 168
  - MULTIVARIATE\_OUTLIER\_CHECK, 168
  - MULTIVARIATE\_OUTLIER\_CHECKTYPE, 169
  - REL\_VAL, 237
  - RELCOMPL\_SPEED, 236
  - RESPT\_PER\_ITEM, 238
  - SCALE\_ACRONYM, 239
  - SCALE\_NAME, 240
  - TOTRESPT, 247
  - VARIABLE\_LIST, 250
  - VARIABLE\_LIST\_ORDER, 251
- \* **options**
  - dataquieR.acc\_loess.exclude\_constant\_subgroups, 65
  - dataquieR.acc\_loess.mark\_time\_points, 66
  - dataquieR.acc\_loess.min\_bw, 67
  - dataquieR.acc\_loess.min\_proportion, 68
  - dataquieR.acc\_loess.plot\_format, 68
  - dataquieR.acc\_loess.plot\_observations, 69
  - dataquieR.acc\_margins\_num, 70
  - dataquieR.acc\_margins\_sort, 71
  - dataquieR.acc\_multivariate\_outlier.scale, 72
  - dataquieR.col\_con\_con\_empirical, 73
  - dataquieR.col\_con\_con\_logical, 74
  - dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, 74
  - dataquieR.CONDITIONS\_WITH\_STACKTRACE, 75
  - dataquieR.convert\_to\_list\_for\_lapply, 76
  - dataquieR.debug, 77
  - dataquieR.des\_summary\_hard\_lim\_remove, 77
  - dataquieR.dontwrapresults, 78
  - dataquieR.droplevels\_ReportSummaryTable, 79
  - dataquieR.dt\_adjust, 80
  - dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, 81
  - dataquieR.ERRORS\_WITH\_CALLER, 82
  - dataquieR.fix\_column\_type\_on\_read, 82
  - dataquieR.flip\_mode, 83
  - dataquieR.force\_item\_specific\_missing\_codes, 84
  - dataquieR.force\_label\_col, 85
  - dataquieR.GAM\_for\_LOESS, 86
  - dataquieR.grading\_formats, 86
  - dataquieR.grading\_rulesets, 87
  - dataquieR.guess\_character, 88
  - dataquieR.guess\_missing\_codes, 89
  - dataquieR.ignore\_empty\_vars, 90
  - dataquieR.lang, 91
  - dataquieR.lazy\_plots, 92
  - dataquieR.lazy\_plots\_cache, 92
  - dataquieR.lazy\_plots\_gg\_compatibility, 93
  - dataquieR.locale, 94
  - dataquieR.MAHALANOBIS\_THRESHOLD, 95
  - dataquieR.max\_cat\_resp\_var\_levels\_in\_plot, 96

- dataquieR.max\_group\_var\_levels\_in\_plot, 96
- dataquieR.max\_group\_var\_levels\_with\_violin, 97
- dataquieR.MAX\_LABEL\_LEN, 98
- dataquieR.MAX\_LONG\_LABEL\_LEN, 99
- dataquieR.MAX\_VALUE\_LABEL\_LEN, 99
- dataquieR.MESSAGES\_WITH\_CALLER, 100
- dataquieR.min\_obs\_per\_group\_var\_in\_plot, 101
- dataquieR.min\_time\_points\_for\_cat\_resp\_var, 102
- dataquieR.MULTIVARIATE\_OUTLIER\_CHECK, 103
- dataquieR.non\_disclosure, 104
- dataquieR.old\_factor\_handling, 104
- dataquieR.old\_type\_adjust, 105
- dataquieR.precomputeStudyData, 106
- dataquieR.print\_block\_load\_factor, 107
- dataquieR.progress\_fkt\_default, 108
- dataquieR.progress\_msg\_fkt\_default, 108
- dataquieR.resume\_checkpoint, 109
- dataquieR.resume\_print, 110
- dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit, 111
- dataquieR.scale\_level\_heuristics\_control\_metriclevels, 111
- dataquieR.study\_data\_cache\_max, 112
- dataquieR.study\_data\_cache\_metrics, 113
- dataquieR.study\_data\_cache\_metrics\_env, 114
- dataquieR.study\_data\_cache\_quick\_fill, 115
- dataquieR.study\_data\_colnames\_case\_sensitive, 116
- dataquieR.testdebug, 117
- dataquieR.traceback, 118
- dataquieR.type\_adjust\_parallel, 118
- dataquieR.VALUE\_LABELS\_htmlescaped, 119
- dataquieR.WARNINGS\_WITH\_CALLER, 120
- progress\_init\_fkt, 233
- study\_data\_cache**
  - dataquieR.ignore\_empty\_vars, 90
  - dataquieR.precomputeStudyData, 106
  - dataquieR.print\_block\_load\_factor, 107
  - dataquieR.study\_data\_cache\_max, 112
  - dataquieR.study\_data\_cache\_metrics, 113
  - dataquieR.study\_data\_cache\_metrics\_env, 114
  - dataquieR.study\_data\_cache\_metrics\_env\_default, 115
  - dataquieR.study\_data\_cache\_quick\_fill, 115
- \* summary\_functions**
  - prep\_combine\_report\_summaries, 185
  - prep\_extract\_classes\_by\_functions, 194
  - prep\_extract\_summary, 194
  - prep\_extract\_summary.dataquieR\_result, 195
  - prep\_extract\_summary.dataquieR\_resultset2, 196
  - prep\_render\_pie\_chart\_from\_summaryclasses\_ggplot2, 217
  - prep\_render\_pie\_chart\_from\_summaryclasses\_plotly, 218
  - prep\_summary\_to\_classes, 222
  - \*.numeric\_with\_unit, 9
  - +.numeric\_with\_unit, 9
  - .numeric\_with\_unit, 10
  - .dataquieR\_resultset2
    - (dataquieR\_resultset2-class), 121
    - /.numeric\_with\_unit, 10
    - [.dataquieR\_resultset2, 254
    - [<-.dataquieR\_resultset2, 255
    - [|.dataquieR\_resultset2, 256
    - [<|-.dataquieR\_resultset2, 256
    - \$.dataquieR\_resultset2, 257
    - \$<-.dataquieR\_resultset2, 257
    - %/.numeric\_with\_unit, 258
    - %|.numeric\_with\_unit, 258
    - ^.numeric\_with\_unit, 259
  - acc\_cat\_distributions, 11

- acc\_distributions, [12](#), [16](#), [18](#), [20](#)
- acc\_distributions\_ecdf, [14](#)
- acc\_distributions\_loc, [15](#)
- acc\_distributions\_only, [17](#)
- acc\_distributions\_prop, [18](#)
- acc\_end\_digits, [20](#)
- acc\_loess, [21](#)
- acc\_loess(), [68](#)
- acc\_mahalanobis, [24](#), [165](#)
- acc\_mahalanobis(), [95](#)
- acc\_margins, [25](#)
- acc\_multivariate\_outlier, [28](#), [168](#), [169](#)
- acc\_robust\_univariate\_outlier, [30](#), [36](#)
- acc\_shape\_or\_scale, [20](#), [32](#), [136](#)
- acc\_univariate\_outlier, [32](#), [34](#)
- acc\_varcomp, [36](#)
- as.character.interval, [38](#)
- as.data.frame.dataquieR\_resultset, [39](#), [121](#), [140](#)
- as.integer(), [104](#)
- as.list.dataquieR\_resultset, [39](#), [121](#), [140](#)
- as.list.dataquieR\_resultset2, [40](#)
- ASSOCIATION\_DIRECTION, [40](#), [41–43](#), [46](#), [55](#), [56](#), [122](#), [147](#), [165–169](#), [236–240](#), [247](#), [251](#)
- ASSOCIATION\_FORM, [40](#), [41](#), [42](#), [43](#), [46](#), [55](#), [56](#), [122](#), [147](#), [165–169](#), [236–240](#), [247](#), [251](#)
- ASSOCIATION\_METRIC, [40](#), [41](#), [41](#), [42](#), [43](#), [46](#), [55](#), [56](#), [122](#), [147](#), [165–169](#), [236–240](#), [247](#), [251](#)
- ASSOCIATION\_RANGE, [40–42](#), [42](#), [43](#), [46](#), [55](#), [56](#), [122](#), [147](#), [165–169](#), [236–240](#), [247](#), [251](#)
- base::rbind.data.frame, [236](#)
- browser(), [77](#)
- cash-.dataquieR\_resultset2  
  (\$.dataquieR\_resultset2), [257](#)
- cash-set-.dataquieR\_resultset2.Rd  
  (\$<-.dataquieR\_resultset2), [257](#)
- cause\_label\_df, [47](#), [174](#), [193](#), [210](#), [249](#)
- cause\_label\_df (value/missing-lists), [249](#)
- character, [11](#), [13](#), [14](#), [16](#), [17](#), [19](#), [20](#), [23](#), [24](#), [27](#), [29](#), [31](#), [33](#), [35](#), [37](#), [48–50](#), [52](#), [53](#), [57](#), [59](#), [60](#), [62](#), [64](#), [80](#), [90](#), [92–94](#), [104](#), [105](#), [116](#), [118](#), [125–127](#), [129](#), [138–140](#), [142](#), [143](#), [150](#), [152–159](#), [161–164](#), [173](#), [174](#), [177–182](#), [184](#), [186–189](#), [191–193](#), [197](#), [198](#), [200–202](#), [206](#), [208](#), [216](#), [220](#), [224](#), [226](#), [235](#), [249](#)
- character(), [204](#), [205](#)
- CHECK\_ID, [40–42](#), [42](#), [43](#), [46](#), [55](#), [56](#), [122](#), [147](#), [165–169](#), [236–240](#), [247](#), [251](#)
- CHECK\_LABEL, [40–43](#), [43](#), [46](#), [55](#), [56](#), [122](#), [147](#), [165–169](#), [236–240](#), [247](#), [251](#)
- check\_table, [44](#), [167](#)
- CO\_VARS, [253](#)
- CO\_VARS  
  (WELL\_KNOWN\_META\_VARIABLE\_NAMES), [252](#)
- CODE\_CLASS (value/missing-lists), [249](#)
- CODE\_CLASSES, [44](#)
- CODE\_INTERPRET (value/missing-lists), [249](#)
- CODE\_LABEL (value/missing-lists), [249](#)
- CODE\_LIST\_TABLE, [44](#)
- CODE\_ORDER, [45](#)
- CODE\_VALUE (value/missing-lists), [249](#)
- com\_item\_missingness, [46](#), [49](#), [50](#)
- com\_item\_missingness(), [249](#)
- com\_qualified\_item\_missingness, [48](#)
- com\_qualified\_item\_missingness(), [249](#)
- com\_qualified\_segment\_missingness, [49](#)
- com\_qualified\_segment\_missingness(), [249](#)
- com\_segment\_missingness, [50](#)
- com\_segment\_missingness(), [249](#)
- com\_unit\_missingness, [53](#)
- COMPATIBILITY (VARATT\_REQUIRE\_LEVELS), [250](#)
- COMPUTATION\_RULE, [45](#)
- COMPUTED\_VARIABLE\_ROLE, [254](#)
- COMPUTED\_VARIABLE\_ROLE  
  (WELL\_KNOWN\_META\_VARIABLE\_NAMES), [252](#)
- COMPUTED\_VARIABLE\_ROLES, [40–43](#), [46](#), [55](#), [56](#), [122](#), [147](#), [165–169](#), [236–240](#), [247](#), [251](#)
- con\_contradictions, [44](#), [56](#)
- con\_contradictions\_recap, [44](#), [58](#)
- con\_inadmissible\_categorical, [60](#)
- con\_inadmissible\_categorical(), [249](#)

- con\_inadmissible\_vocabulary, 61  
 con\_inadmissible\_vocabulary(), 249  
 con\_limit\_deviations, 63  
 contradiction\_functions\_descriptions, 54  
 CONTRADICTION\_TERM, 40–43, 46, 55, 56, 103, 122, 147, 165–169, 236–240, 247, 250, 251  
 CONTRADICTION\_TYPE, 40–43, 46, 55, 55, 122, 147, 165–169, 236–240, 247, 251  
 CONTRADICTIONS, 253  
 CONTRADICTIONS  
   (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252  
  
 data.frame, 11–17, 19, 20, 22, 24–33, 35–37, 47–54, 56–60, 62, 64, 65, 86, 87, 125–129, 138, 140, 142–144, 150, 152–159, 161–164, 173–182, 184, 186–188, 190, 192–198, 200, 201, 208–214, 220, 222–224, 234, 235, 249  
 DATA\_ENTRY\_TYPE, 253  
 DATA\_ENTRY\_TYPE  
   (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252  
 DATA\_PREPARATION, 40–43, 46, 55, 56, 58, 122, 147, 165–169, 236–240, 247, 251  
 DATA\_TYPE, 88, 123, 252  
 DATA\_TYPE  
   (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252  
 DATA\_TYPES, 123, 252  
 DATA\_TYPES\_OF\_R\_TYPE, 124, 191  
 DATAFRAMES, 254  
 DATAFRAMES  
   (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252  
 dataquieR, 30, 34, 65–120, 123, 225, 226, 233, 246  
 dataquieR report v2, 226  
 dataquieR.acc\_loess.exclude\_constant\_subgroups, 65, 66–120, 233  
 dataquieR.acc\_loess.mark\_time\_points, 65, 66, 67–120, 233  
 dataquieR.acc\_loess.min\_bw, 65, 66, 67, 68–120, 233  
 dataquieR.acc\_loess.min\_proportion, 65–67, 68, 69–120, 233  
 dataquieR.acc\_loess.plot\_format, 65–68, 68, 69–120, 233  
 dataquieR.acc\_loess.plot\_observations, 66–69, 69, 70–120, 233  
 dataquieR.acc\_margins\_num, 66–69, 70, 71–120, 233  
 dataquieR.acc\_margins\_sort, 66–70, 71, 72–114, 116–120, 233  
 dataquieR.acc\_multivariate\_outlier.scale, 66–71, 72, 73–114, 116–120, 233  
 dataquieR.applicability\_problem, 72  
 dataquieR.col\_con\_con\_empirical, 66–72, 73, 74–114, 116–120, 233  
 dataquieR.col\_con\_con\_logical, 66–73, 74, 75–114, 116–120, 233  
 dataquieR.CONDITIONS\_LEVEL\_TRHESHOLD, 65–74, 74, 75–120, 233  
 dataquieR.CONDITIONS\_WITH\_STACKTRACE, 65–75, 75, 76–120, 233  
 dataquieR.convert\_to\_list\_for\_lapply, 66–75, 76, 77–114, 116–120, 233  
 dataquieR.debug, 66–76, 77, 78–114, 116–120, 233  
 dataquieR.des\_summary\_hard\_lim\_remove, 66–77, 77, 79–114, 116–120, 233  
 dataquieR.dontwrapresults, 66–78, 78, 79–114, 116–120, 233  
 dataquieR.droplevels\_ReportSummaryTable, 66–79, 79, 80–114, 116–120, 231, 233  
 dataquieR.dt\_adjust, 66–79, 80, 81–114, 116–120, 140, 233  
 dataquieR.ELEMENT\_MISMATCH\_CHECKTYPE, 65–80, 81, 82–120, 158, 233  
 dataquieR.ERRORS\_WITH\_CALLER, 65–81, 82, 83–120, 233  
 dataquieR.fix\_column\_type\_on\_read, 66–82, 82, 84–114, 116–120, 233  
 dataquieR.flip\_mode, 66–83, 83, 84–114, 116–120, 233  
 dataquieR.force\_item\_specific\_missing\_codes, 66–84, 84, 85–114, 116–120, 233  
 dataquieR.force\_label\_col, 66–84, 85, 86–114, 116–120, 233  
 dataquieR.GAM\_for\_LOESS, 65–85, 86, 87–120, 233

- dataquieR.grading\_formats, [66–86](#), [86](#),  
[88–114](#), [116–120](#), [233](#)
- dataquieR.grading\_rulesets, [66–87](#), [87](#),  
[88–114](#), [116–120](#), [233](#)
- dataquieR.guess\_character, [66–88](#), [88](#),  
[89–114](#), [116–120](#), [233](#)
- dataquieR.guess\_missing\_codes, [66–88](#),  
[89](#), [90–114](#), [116–120](#), [233](#)
- dataquieR.ignore\_empty\_vars, [66–89](#), [90](#),  
[91–120](#), [139](#), [233](#)
- dataquieR.intrinsic\_applicability\_problem,  
[90](#)
- dataquieR.lang, [66–90](#), [91](#), [92–114](#),  
[116–120](#), [234](#)
- dataquieR.lazy\_plots, [66–91](#), [92](#), [93–114](#),  
[116–120](#), [234](#)
- dataquieR.lazy\_plots\_cache, [66–92](#), [92](#),  
[94–114](#), [116–120](#), [216](#), [234](#)
- dataquieR.lazy\_plots\_gg\_compatibility,  
[66–93](#), [93](#), [94–114](#), [116–120](#), [234](#)
- dataquieR.locale, [66–94](#), [94](#), [95–114](#),  
[116–120](#), [234](#)
- dataquieR.MAHALANOBIS\_THRESHOLD, [65–94](#),  
[95](#), [96–120](#), [233](#)
- dataquieR.max\_cat\_resp\_var\_levels\_in\_plot,  
[66–95](#), [96](#), [97–114](#), [116–120](#), [234](#)
- dataquieR.max\_group\_var\_levels\_in\_plot,  
[66–96](#), [96](#), [98–114](#), [116–120](#), [234](#)
- dataquieR.max\_group\_var\_levels\_with\_violins,  
[66–97](#), [97](#), [98–114](#), [116–120](#), [234](#)
- dataquieR.MAX\_LABEL\_LEN, [65–97](#), [98](#),  
[99–120](#), [233](#)
- dataquieR.MAX\_LONG\_LABEL\_LEN, [65–98](#), [99](#),  
[100–120](#), [233](#)
- dataquieR.MAX\_VALUE\_LABEL\_LEN, [65–99](#),  
[99](#), [100–120](#), [233](#)
- dataquieR.MESSAGES\_WITH\_CALLER, [65–100](#),  
[100](#), [101–120](#), [233](#)
- dataquieR.min\_obs\_per\_group\_var\_in\_plot,  
[66–101](#), [101](#), [102–114](#), [116–120](#),  
[234](#)
- dataquieR.min\_time\_points\_for\_cat\_resp\_var,  
[66–101](#), [102](#), [103–114](#), [116–120](#),  
[234](#)
- dataquieR.MULTIVARIATE\_OUTLIER\_CHECK,  
[65–102](#), [103](#), [104–120](#), [165](#), [169](#),  
[233](#)
- dataquieR.non\_disclosure, [66–103](#), [104](#),  
[105–114](#), [116–120](#), [234](#)
- dataquieR.old\_factor\_handling, [66–104](#),  
[104](#), [106–114](#), [116–120](#), [234](#)
- dataquieR.old\_type\_adjust, [66–105](#), [105](#),  
[106–114](#), [116–120](#), [234](#)
- dataquieR.precomputeStudyData, [66–106](#),  
[106](#), [107–120](#), [234](#)
- dataquieR.print\_block\_load\_factor,  
[66–107](#), [107](#), [108–120](#), [226](#), [234](#)
- dataquieR.progress\_fkt\_default, [66–107](#),  
[108](#), [109–114](#), [116–120](#), [234](#)
- dataquieR.progress\_msg\_fkt\_default,  
[66–108](#), [108](#), [110–114](#), [116–120](#),  
[234](#)
- dataquieR.resume\_checkpoint, [66–109](#),  
[109](#), [110–114](#), [116–120](#), [234](#)
- dataquieR.resume\_print, [66–110](#), [110](#),  
[111–114](#), [116–120](#), [234](#)
- dataquieR.scale\_level\_heuristics\_control\_binaryrecodelimit,  
[66–110](#), [111](#), [112–114](#), [116–120](#),  
[234](#)
- dataquieR.scale\_level\_heuristics\_control\_metriclevels,  
[66–111](#), [111](#), [113](#), [114](#), [116–121](#),  
[234](#)
- dataquieR.study\_data\_cache\_max, [66–112](#),  
[112](#), [113–121](#), [234](#)
- dataquieR.study\_data\_cache\_metrics,  
[66–113](#), [113](#), [114–121](#), [234](#)
- dataquieR.study\_data\_cache\_metrics\_env,  
[66–114](#), [114](#), [115–121](#), [234](#)
- dataquieR.study\_data\_cache\_metrics\_env\_default,  
[90](#), [107](#), [113–115](#), [115](#), [116](#)
- dataquieR.study\_data\_cache\_quick\_fill,  
[66–115](#), [115](#), [117–121](#), [234](#)
- dataquieR.study\_data\_colnames\_case\_sensitive,  
[66–114](#), [116](#), [116](#), [117–121](#), [234](#)
- dataquieR.testdebug, [66–114](#), [116](#), [117](#),  
[117](#), [118–121](#), [234](#)
- dataquieR.traceback, [66–114](#), [116](#), [117](#),  
[118](#), [119–121](#), [234](#)
- dataquieR.type\_adjust\_parallel, [66–114](#),  
[116–118](#), [118](#), [120](#), [121](#), [234](#)
- dataquieR.VALUE\_LABELS\_htmlescaped,  
[65–119](#), [119](#), [120](#), [233](#)
- dataquieR.WARNINGS\_WITH\_CALLER, [65–119](#),  
[120](#), [233](#)
- dataquieR\_result, [194](#), [195](#), [229](#)
- dataquieR\_result

- (print.dataquieR\_result), 225
- dataquieR\_resultset, 121, 121
- dataquieR\_resultset2, 40, 121, 140, 189, 206, 220, 221, 257
- dataquieR\_resultset2
  - (dataquieR\_resultset2-class), 121
- dataquieR\_resultset2-class, 121
- dataquieR\_resultset\_verify, 122
- DATETIME, 249
- DATETIME (DATA\_TYPES), 123
- datetime (DATA\_TYPES), 123
- DECIMALS, 253
- DECIMALS
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- default.stringsAsFactors, 187
- des\_scatterplot\_matrix, 124
- des\_summary, 125
- des\_summary\_categorical, 127
- des\_summary\_continuous, 128
- Descriptor, 11, 14, 17, 21, 51, 53, 56, 125, 127, 128, 158
- DETECTION\_LIMIT\_LOW, 253
- DETECTION\_LIMIT\_LOW
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- DETECTION\_LIMIT\_UP, 253
- DETECTION\_LIMIT\_UP
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- DETECTION\_LIMITS, 253
- DETECTION\_LIMITS
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- DF\_CODE, 130
- DF\_ELEMENT\_COUNT, 130
- DF\_ID\_REF\_TABLE, 131
- DF\_ID\_VARS, 131
- DF\_NAME, 132
- DF\_RECORD\_CHECK, 132
- DF\_RECORD\_COUNT, 133
- DF\_UNIQUE\_ID, 133, 244
- DF\_UNIQUE\_ROWS, 134
- dim.dataquieR\_resultset2, 134
- dimensions, 135, 138
- dimnames.dataquieR\_resultset2, 135
- dims, 136
- DISTRIBUTION, 253
- DISTRIBUTION
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- DISTRIBUTIONS, 136
- dq\_lazy\_ggplot\_methods
  - (ggplotGrob.dq\_lazy\_ggplot), 145
- dq\_report, 121, 137, 143, 144
- dq\_report2, 121, 137, 143, 144, 194–196, 210, 225
- dq\_report2(), 118, 172, 205, 227
- dq\_report\_by, 140, 141, 251
- droplevels.ReportSummaryTable, 145
- emmeans::emmeans, 25
- ENCODING, 254
- ENCODING
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- END\_DIGIT\_CHECK, 253
- END\_DIGIT\_CHECK
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- enum, 12, 13, 15–17, 19, 22, 26, 33, 47, 49, 50, 52, 64, 139, 157, 158, 182, 187, 200, 222, 231, 235, 249
- enum (DATA\_TYPES), 123
- environment, 114, 198
- eval.parent, 214
- EXTENDED\_DATA\_TYPE, 254
- EXTENDED\_DATA\_TYPE
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- FLOAT (DATA\_TYPES), 123
- float, 123
- float (DATA\_TYPES), 123
- function, 108, 140, 143, 212, 233
- ggplot2::geom\_jitter, 31, 34
- ggplot2::geom\_line(), 22
- ggplot2::ggplot, 12, 13, 15–17, 19, 32, 36, 57, 59, 65, 125
- ggplot2::ggplot(), 28
- ggplot2::ggplot2, 25, 30, 33, 154, 235
- ggplotGrob, 146
- ggplotGrob.dq\_lazy\_ggplot, 145

- ggplotGrob.dq\_lazy\_ggplot\_s7  
(ggplotGrob.dq\_lazy\_ggplot),  
[145](#)
- ggplotly.dq\_lazy\_ggplot  
(ggplotGrob.dq\_lazy\_ggplot),  
[145](#)
- ggplotly.dq\_lazy\_ggplot\_s7  
(ggplotGrob.dq\_lazy\_ggplot),  
[145](#)
- glm, [184](#)
- GOLDSTANDARD, [40–43](#), [46](#), [55](#), [56](#), [122](#), [146](#),  
[165–169](#), [236–240](#), [247](#), [251](#)
- GRADING\_RULESET, [87](#), [254](#)
- GRADING\_RULESET  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- grid.draw.util\_pairs\_ggplot\_panels,  
[147](#)
- GROUP\_VAR\_DEVICE, [253](#)
- GROUP\_VAR\_DEVICE  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- GROUP\_VAR\_OBSERVER, [253](#)
- GROUP\_VAR\_OBSERVER  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- HARD\_LIMIT\_LOW, [253](#)
- HARD\_LIMIT\_LOW  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- HARD\_LIMIT\_UP, [253](#)
- HARD\_LIMIT\_UP  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- HARD\_LIMITS, [253](#)
- HARD\_LIMITS  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- html\_dependency\_clipboard, [147](#)
- html\_dependency\_dataquieR, [148](#)
- html\_dependency\_jspdf, [148](#)
- html\_dependency\_report\_dt, [149](#)
- html\_dependency\_tippy, [149](#)
- html\_dependency\_vert\_dt, [149](#)
- htmltools::htmlDependency(), [148](#)
- INCL\_HARD\_LIMIT\_LOW  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- INCL\_HARD\_LIMIT\_UP, [253](#)
- INCL\_HARD\_LIMIT\_UP  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- INCL\_LOCATION\_LIMIT\_LOW, [254](#)
- INCL\_LOCATION\_LIMIT\_LOW  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- INCL\_LOCATION\_LIMIT\_UP, [254](#)
- INCL\_LOCATION\_LIMIT\_UP  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- INCL\_PROPORTION\_LIMIT\_LOW, [254](#)
- INCL\_PROPORTION\_LIMIT\_LOW  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- INCL\_PROPORTION\_LIMIT\_UP, [254](#)
- INCL\_PROPORTION\_LIMIT\_UP  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- INCL\_SOFT\_LIMIT\_LOW, [253](#)
- INCL\_SOFT\_LIMIT\_LOW  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- INCL\_SOFT\_LIMIT\_UP, [253](#)
- INCL\_SOFT\_LIMIT\_UP  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- Indicator, [12](#), [15](#), [18](#), [20](#), [24](#), [25](#), [28](#), [31](#), [32](#),  
[34](#), [36](#), [46](#), [48](#), [49](#), [58](#), [60](#), [61](#), [63](#),  
[150–152](#), [154–156](#), [159–162](#), [164](#)
- int\_all\_datastructure\_dataframe, [150](#)
- int\_all\_datastructure\_segment, [151](#)
- int\_datatype\_matrix, [152](#)
- int\_duplicate\_content, [154](#)
- int\_duplicate\_ids, [155](#)
- int\_encoding\_errors, [156](#)
- int\_part\_vars\_structure, [157](#)
- int\_sts\_element\_dataframe, [158](#)
- int\_sts\_element\_segment, [159](#)
- int\_unexp\_elements, [160](#)
- int\_unexp\_records\_dataframe, [161](#)
- int\_unexp\_records\_segment, [162](#)
- int\_unexp\_records\_set, [164](#)
- INTEGER (DATA\_TYPES), [123](#)

- integer, [22](#), [23](#), [26](#), [27](#), [29](#), [31](#), [35](#), [37](#), [123](#),  
[139](#), [153](#), [161–163](#), [200](#), [211](#), [219](#),  
[220](#), [227](#), [234](#), [235](#)
- integer (DATA\_TYPES), [123](#)
- invisible(), [187](#)
- IRV, [40–43](#), [46](#), [55](#), [56](#), [122](#), [147](#), [165](#),  
[166–169](#), [236–240](#), [247](#), [251](#)
- JUMP\_LIST, [84](#), [252](#)
- JUMP\_LIST  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- KEY\_DATETIME, [253](#)
- KEY\_DATETIME  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- KEY\_DEVICE, [253](#)
- KEY\_DEVICE  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- KEY\_OBSERVER, [253](#)
- KEY\_OBSERVER  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- KEY\_STUDY\_SEGMENT, [253](#)
- KEY\_STUDY\_SEGMENT  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- LABEL, [98](#), [250](#), [252](#)
- LABEL (WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- list, [12](#), [13](#), [15–17](#), [19](#), [20](#), [22](#), [23](#), [40](#), [49](#), [50](#),  
[57](#), [59](#), [65](#), [126](#), [128](#), [129](#), [139](#), [143](#),  
[144](#), [150](#), [152](#), [154](#), [155](#), [159–164](#),  
[186](#), [193](#), [195](#), [196](#), [208](#), [209](#), [212](#),  
[224–226](#), [235](#)
- list(), [222](#), [229](#)
- list.files(), [205](#)
- lme4::lmer, [184](#)
- LOCATION\_LIMIT\_LOW, [253](#)
- LOCATION\_LIMIT\_LOW  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- LOCATION\_LIMIT\_UP, [253](#)
- LOCATION\_LIMIT\_UP  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- LOCATION\_METRIC, [253](#)
- LOCATION\_METRIC  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- LOCATION\_RANGE, [253](#)
- LOCATION\_RANGE  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- logical, [13](#), [15](#), [19](#), [22](#), [23](#), [27](#), [29](#), [33](#), [47](#), [48](#),  
[57–59](#), [64](#), [126](#), [127](#), [129](#), [139](#), [140](#),  
[143](#), [148](#), [153](#), [158](#), [172](#), [174](#), [175](#),  
[177](#), [179](#), [182](#), [184](#), [186–188](#),  
[190–192](#), [198](#), [200](#), [205](#), [207](#), [208](#),  
[210](#), [213](#), [214](#), [221](#), [222](#), [226](#), [227](#),  
[231](#), [234](#)
- LONG\_LABEL, [99](#), [250](#), [253](#)
- LONG\_LABEL  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- MAHALANOBIS\_THRESHOLD, [40–43](#), [46](#), [55](#), [56](#),  
[122](#), [147](#), [165](#), [165](#), [166–169](#),  
[236–240](#), [247](#), [251](#)
- match.arg, [123](#)
- MAXIMUM\_LONG\_STRING, [40–43](#), [46](#), [55](#), [56](#),  
[122](#), [147](#), [165](#), [166](#), [166](#), [167–169](#),  
[236–240](#), [247](#), [251](#)
- meta\_data, [157](#), [166](#), [197](#), [217](#), [218](#)
- meta\_data\_computation, [40–43](#), [45](#), [46](#), [55](#),  
[56](#), [122](#), [147](#), [165–167](#), [167](#), [168](#),  
[169](#), [236–240](#), [247](#), [251](#)
- meta\_data\_cross, [40–44](#), [46](#), [55](#), [56](#), [60](#), [64](#),  
[122](#), [125](#), [147](#), [165–167](#), [167](#), [168](#),  
[169](#), [236–240](#), [247](#), [251](#)
- meta\_data\_dataframe, [130–134](#), [166](#), [167](#)
- meta\_data\_segment, [166](#), [168](#), [241–245](#), [254](#)
- mget, [208](#)
- MISS\_RESP, [40–43](#), [46](#), [55](#), [56](#), [122](#), [147](#),  
[165–167](#), [168](#), [169](#), [236–240](#), [247](#),  
[251](#)
- MISSING\_LIST, [84](#), [252](#)
- MISSING\_LIST  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)
- MISSING\_LIST\_TABLE, [84](#), [193](#), [249](#), [252](#)
- MISSING\_LIST\_TABLE  
(WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
[252](#)

- missing\_matchtable
  - (value/missing-lists), 249
- MULTIVARIATE\_OUTLIER\_CHECK, 40–43, 46, 55, 56, 122, 147, 165–168, 168, 169, 236–240, 247, 251
- MULTIVARIATE\_OUTLIER\_CHECKTYPE, 40–43, 46, 55, 56, 122, 147, 165–169, 169, 236–240, 247, 251
- N\_RULES, 169, 254
- N\_RULES
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- nres, 170
- numeric, 22–24, 26, 29, 33, 47, 51, 56, 58, 60, 62, 107, 130, 133, 153, 219, 226, 244, 249
- numeric (DATA\_TYPER), 123
- OPTIONAL (VARATT\_REQUIRE\_LEVELS), 250
- options(), 139, 143, 226
- parallelMap::parallelStart, 139, 212, 227
- PART\_VAR, 253
- PART\_VAR
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- pipeline\_recursive\_result, 170
- pipeline\_vectorized, 171
- plot.dataquieR\_summary, 171
- plotly\_build.dq\_lazy\_ggplot
  - (ggplotGrob.dq\_lazy\_ggplot), 145
- plotly\_build.dq\_lazy\_ggplot\_s7
  - (ggplotGrob.dq\_lazy\_ggplot), 145
- prep\_acc\_distributions\_with\_ecdf, 172
- prep\_add\_cause\_label\_df, 173, 193
- prep\_add\_computed\_variables, 174
- prep\_add\_data\_frames, 175, 198, 203, 205, 207, 216, 217
- prep\_add\_data\_frames(), 193
- prep\_add\_missing\_codes, 176
- prep\_add\_to\_meta, 177
- prep\_apply\_coding, 178
- prep\_check\_for\_dataquieR\_updates, 179
- prep\_check\_meta\_data\_dataframe, 180
- prep\_check\_meta\_data\_segment, 181
- prep\_check\_meta\_names, 182, 188
- prep\_clean\_labels, 184
- prep\_combine\_report\_summaries, 185, 194–196, 218, 223
- prep\_combine\_report\_summaries(), 195, 196
- prep\_compare\_meta\_with\_study, 186
- prep\_create\_meta, 178, 187, 250
- prep\_create\_meta\_data\_file, 188
- prep\_create\_storr\_factory, 189
- prep\_create\_storr\_factory(), 207
- prep\_datatype\_from\_data, 189
- prep\_datatype\_from\_data(), 219
- prep\_deparse\_assignments, 190
- prep\_dq\_data\_type\_of, 124, 191
- prep\_expand\_codes, 192
- prep\_extract\_cause\_label\_df, 174, 193
- prep\_extract\_classes\_by\_functions, 186, 194, 195, 196, 218, 223
- prep\_extract\_summary, 186, 194, 194, 195, 196, 218, 223
- prep\_extract\_summary(), 222
- prep\_extract\_summary.dataquieR\_result, 186, 194, 195, 195, 196, 218, 223
- prep\_extract\_summary.dataquieR\_resultset2, 186, 194, 195, 196, 218, 223
- prep\_fix\_meta\_id\_dups, 196
- prep\_get\_data\_frame, 176, 198, 203, 205, 207, 216, 217
- prep\_get\_data\_frame(), 83, 204, 205
- prep\_get\_labels, 199
- prep\_get\_study\_data\_segment, 201
- prep\_get\_user\_name, 202
- prep\_guess\_encoding, 202
- prep\_link\_escape, 203
- prep\_list\_dataframes, 176, 198, 203, 205, 207, 216, 217
- prep\_list\_voc, 204
- prep\_load\_folder\_with\_metadata, 176, 198, 203, 205, 207, 216, 217
- prep\_load\_report, 206
- prep\_load\_report\_from\_backend, 206
- prep\_load\_workbook\_like\_file, 11, 13, 14, 16, 17, 19, 20, 23, 24, 27, 29, 31, 33, 35, 37, 48–50, 52, 53, 57, 59, 60, 62, 64, 125–127, 129, 138, 142, 150, 152–156, 158, 159, 161–164, 173, 174, 176–182, 184, 186, 192, 193,

- [197, 198, 200, 201, 203, 205, 207, 208, 216, 217, 220, 235](#)
- [prep\\_map\\_labels, 123, 208](#)
- [prep\\_merge\\_study\\_data, 209](#)
- [prep\\_meta\\_data\\_v1\\_to\\_item\\_level\\_meta\\_data, 210](#)
- [prep\\_min\\_obs\\_level, 211](#)
- [prep\\_open\\_in\\_excel, 211](#)
- [prep\\_pmap, 212](#)
- [prep\\_prepare\\_dataframes, 213](#)
- [prep\\_purge\\_data\\_frame\\_cache, 11, 13, 14, 16, 17, 19, 20, 23, 24, 27, 29, 31, 33, 35, 37, 48–50, 52, 53, 57, 59, 60, 62, 64, 125–127, 129, 138, 142, 150, 152–156, 158, 159, 161–164, 173, 174, 176–182, 184, 186, 192, 193, 197, 198, 200, 201, 203, 205, 207, 208, 215, 217, 220, 235](#)
- [prep\\_purge\\_data\\_frame\\_cache\(\), 205](#)
- [prep\\_realize\\_ggplot, 216](#)
- [prep\\_realize\\_ggplot\(\), 93](#)
- [prep\\_remove\\_from\\_cache, 176, 198, 203, 205, 207, 216, 216](#)
- [prep\\_remove\\_from\\_cache\(\), 205](#)
- [prep\\_render\\_pie\\_chart\\_from\\_summaryclasses\\_ggplot2, 186, 194–196, 217, 218, 223](#)
- [prep\\_render\\_pie\\_chart\\_from\\_summaryclasses\\_plotly, 186, 194–196, 218, 218, 223](#)
- [prep\\_robust\\_guess\\_data\\_type, 219](#)
- [prep\\_robust\\_guess\\_data\\_type\(\), 190](#)
- [prep\\_save\\_report, 219](#)
- [prep\\_scalelevel\\_from\\_data\\_and\\_metadata, 220](#)
- [prep\\_scalelevel\\_from\\_data\\_and\\_metadata\(\), 213](#)
- [prep\\_set\\_backend, 221](#)
- [prep\\_set\\_backend\(\), 40](#)
- [prep\\_study2meta, 214, 221](#)
- [prep\\_summary\\_to\\_classes, 186, 194–196, 218, 222](#)
- [prep\\_summary\\_to\\_classes\(\), 194](#)
- [prep\\_title\\_escape, 223](#)
- [prep\\_undisclose, 223](#)
- [prep\\_unsplit\\_val\\_tabs, 224](#)
- [prep\\_valuelabels\\_from\\_data, 224](#)
- [print.dataquieR\\_result, 225](#)
- [print.dataquieR\\_resultset, 121, 140, 225](#)
- [print.dataquieR\\_resultset2, 226](#)
- [print.dataquieR\\_resultset2\(\), 143](#)
- [print.dataquieR\\_summary, 227](#)
- [print.DataSlot, 228](#)
- [print.interval, 228](#)
- [print.list, 229](#)
- [print.master\\_result, 229](#)
- [print.numeric\\_with\\_unit, 230](#)
- [print.ReportSummaryTable, 230](#)
- [print.Slot, 231](#)
- [print.StudyDataSlot, 232](#)
- [print.TableSlot, 232](#)
- [print.util\\_pairs\\_ggplot\\_panels, 233](#)
- [printed, 140](#)
- [pro\\_applicability\\_matrix, 234](#)
- [progress\\_init\\_fkt, 66–114, 116–121, 233](#)
- [PROPORTION\\_LIMIT\\_LOW, 254](#)
- [PROPORTION\\_LIMIT\\_LOW \(WELL\\_KNOWN\\_META\\_VARIABLE\\_NAMES\), 252](#)
- [PROPORTION\\_LIMIT\\_UP, 254](#)
- [PROPORTION\\_LIMIT\\_UP \(WELL\\_KNOWN\\_META\\_VARIABLE\\_NAMES\), 252](#)
- [PROPORTION\\_RANGE, 253](#)
- [PROPORTION\\_RANGE \(WELL\\_KNOWN\\_META\\_VARIABLE\\_NAMES\), 252](#)
- [rbind.ReportSummaryTable, 236](#)
- [readr::guess\\_parser\(\), 219](#)
- [RECODE\\_CASES, 254](#)
- [RECODE\\_CASES \(WELL\\_KNOWN\\_META\\_VARIABLE\\_NAMES\), 252](#)
- [RECODE\\_CONTROL, 254](#)
- [RECODE\\_CONTROL \(WELL\\_KNOWN\\_META\\_VARIABLE\\_NAMES\), 252](#)
- [RECOMMENDED \(VARATT\\_REQUIRE\\_LEVELS\), 250](#)
- [REL\\_VAL, 40–43, 46, 55, 56, 122, 147, 165–169, 236, 237, 237, 238–240, 247, 251](#)
- [RELCOMPL\\_SPEED, 40–43, 46, 55, 56, 122, 147, 165–169, 236, 237–240, 247, 251](#)
- [REQUIRED \(VARATT\\_REQUIRE\\_LEVELS\), 250](#)
- [resnames, 237](#)
- [resnames.dataquieR\\_resultset2, 238](#)
- [RESPT\\_PER\\_ITEM, 40–43, 46, 55, 56, 122, 147, 165–169, 236, 237, 238, 239, 240,](#)

- 247, 251
- robustbase::mc, 30, 34
- RULE (CONTRADICTION\_TERM), 55
- SCALE\_ACRONYM, 40–43, 46, 55, 56, 122, 147, 165–169, 236–238, 239, 240, 247, 251
- SCALE\_LEVEL, 111, 239, 252
- SCALE\_LEVEL
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- SCALE\_LEVELS, 239
- SCALE\_NAME, 40–43, 46, 55, 56, 122, 147, 165–169, 236–239, 240, 247, 251
- SEGMENT\_ID\_REF\_TABLE, 241, 241
- SEGMENT\_ID\_TABLE, 241
- SEGMENT\_ID\_VARS, 242
- SEGMENT\_MISS, 242
- SEGMENT\_PART\_VARS, 243
- SEGMENT\_RECORD\_CHECK, 243
- SEGMENT\_RECORD\_COUNT, 244
- SEGMENT\_UNIQUE\_ID, 244
- SEGMENT\_UNIQUE\_ROWS, 245
- set, 29, 31, 35
- set (DATA\_TYPES), 123
- SOFT\_LIMIT\_LOW, 253
- SOFT\_LIMIT\_LOW
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- SOFT\_LIMIT\_UP, 253
- SOFT\_LIMIT\_UP
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- SOFT\_LIMITS, 253
- SOFT\_LIMITS
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- SPLIT\_CHAR, 245
- STANDARDIZED\_VOCABULARY\_TABLE, 249, 254
- STANDARDIZED\_VOCABULARY\_TABLE
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- STRING (DATA\_TYPES), 123
- string, 123
- string (DATA\_TYPES), 123
- study\_data, 157, 209, 245
- STUDY\_SEGMENT, 142, 195, 196, 253
- STUDY\_SEGMENT
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- 252
- summary(), 172, 227
- summary.dataquieR\_resultset, 121, 140, 246
- summary.dataquieR\_resultset2, 246
- TECHNICAL (VARATT\_REQUIRE\_LEVELS), 250
- TIME (DATA\_TYPES), 123
- time (DATA\_TYPES), 123
- TIME\_VAR, 253
- TIME\_VAR
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- TIME\_VAR\_END, 253
- TIME\_VAR\_END
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- TOTRESPT, 40–43, 46, 55, 56, 122, 147, 165–169, 236–240, 247, 251
- UNIT, 252
- UNIT (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- UNIT\_IS\_COUNT, 247, 248, 248, 249, 254
- UNIT\_PREFIX\_FACTORS, 247, 248, 248, 249, 254
- UNIT\_PREFIXES, 247, 248, 248, 249, 254
- UNIT\_SOURCES, 247, 248, 249, 254
- UNITS, 247, 248, 249, 252, 254
- units::valid\_udunits(), 247–249
- units::valid\_udunits\_prefixes(), 248
- UNIVARIATE\_OUTLIER\_CHECKTYPE, 169, 254
- UNIVARIATE\_OUTLIER\_CHECKTYPE
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- UNKNOWN (VARATT\_REQUIRE\_LEVELS), 250
- value/missing-lists, 249
- VALUE\_LABEL\_TABLE, 249, 252
- VALUE\_LABEL\_TABLE
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- value\_label\_table
  - (value/missing-lists), 249
- VALUE\_LABELS, 119, 252
- VALUE\_LABELS
  - (WELL\_KNOWN\_META\_VARIABLE\_NAMES), 252
- VAR\_NAMES, 175, 195, 196, 200, 250, 252

VAR\_NAMES  
    (WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
    252

VARATT\_REQUIRE\_LEVELS, 182, 187, 214, 222,  
250

variable, 11, 20, 22, 26, 29, 33, 37, 51, 53,  
126, 127, 129, 142, 143, 153, 156,  
190, 224

variable (DATA\_TYPES), 123

variable attribute, 11, 12, 14, 15, 17, 19,  
20, 22, 24, 26, 29, 31, 33, 35, 37, 47,  
49–51, 53, 56, 58, 60, 62, 64, 123,  
125–127, 129, 138, 142, 152–156,  
159, 163, 164, 173–175, 177, 186,  
193, 200, 210, 220, 234, 239

variable attribute  
    (WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
    252

variable list, 12, 14, 15, 17, 19, 22, 24, 26,  
29, 31, 35, 37, 47, 49, 53, 56, 60, 62,  
64, 139, 173, 177, 200, 211, 220

variable list (DATA\_TYPES), 123

variable roles, 31, 35, 52

variable roles (VARIABLE\_ROLES), 251

VARIABLE\_LIST, 40–43, 46, 55, 56, 122, 147,  
165–169, 236–240, 247, 250, 251

VARIABLE\_LIST\_ORDER, 40–43, 46, 55, 56,  
122, 147, 165–169, 236–240, 247,  
251, 251

VARIABLE\_ORDER, 253

VARIABLE\_ORDER  
    (WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
    252

VARIABLE\_ROLE, 253

VARIABLE\_ROLE  
    (WELL\_KNOWN\_META\_VARIABLE\_NAMES),  
    252

VARIABLE\_ROLES, 46, 251

WELL\_KNOWN\_META\_VARIABLE\_NAMES, 187,  
188, 247–249, 252