Package 'REDCapDM'

January 20, 2025

Type Package

Title 'REDCap' Data Management

Version 0.9.9

Maintainer João Carmezim <jcarmezim@igtp.cat>

Description REDCap Data Management - REDCapDM is an R package that allows users to manage data exported directly from REDCap or using an API connection. This package includes several functions designed for pre-processing data, generating reports of queries such as outliers or missing values, and following up on the identified queries. 'REDCap' (Research Electronic Data CAPture; <https://projectredcap.org>) is a web application developed at Vanderbilt University, designed for creating and managing online surveys and databases and the REDCap API is an interface that allows external applications to connect to REDCap remotely, and is used to programmatically retrieve or modify project data or settings within REDCap, such as importing or exporting data.

License MIT + file LICENSE

URL https://bruigtp.github.io/REDCapDM/,
https://doi.org/10.1186/s12874-024-02178-6

BugReports https://github.com/bruigtp/REDCapDM/issues

Encoding UTF-8

RoxygenNote 7.3.1

Imports dplyr, janitor, magrittr, openxlsx, purrr, REDCapR, rlang, stringr, tibble, tidyr, tidyselect, utils, stringi, labelled, cli, forcats

Suggests knitr, rmarkdown, kableExtra

VignetteBuilder knitr

Depends R (>= 3.6)

LazyData true

NeedsCompilation no

Author João Carmezim [aut, cre], Pau Satorra [aut], Judith Peñafiel [aut],

Esther García [aut],
Natàlia Pallarès [aut],
Cristian Tebé [aut]

Repository CRAN

Date/Publication 2024-05-06 12:30:06 UTC

Contents

checkbox_names 2
check_queries 3
covican 4
fill_data 5
rd_event 6
rd_export 7
rd_insert_na 8
rd_query 9
rd_rlogic 11
rd_transform 12
recalculate 13
redcap_data 14
split_event 15
split_form 16
to_factor 16
transform_checkboxes 17

Index **18**

`checkbox_names` *Change checkboxes names into the name of their options*

Description

Function that returns both data and dictionary with the name of the checkboxes transformed by the name of their options.

Usage

```
checkbox_names(data, dic, labels, checkbox_labels = c("No", "Yes"))
```

Arguments

`data`	Dataset containing the REDCap data.
`dic`	Dataset containing the REDCap dictionary.
`labels`	Named character vector with the name of the variables in the data and the REDCap label in its name.

checkbox_labels
: Character vector with the names that will have the two options of every checkbox variable. Default is c('No', 'Yes').

---

check_queries *Check for changes between two reports of the queries*

Description

This function compares a previous report of queries with a new one and allows you to check which queries are new, which have been modified, and which remain unchanged.

Usage

```
check_queries(old, new, report_title = NULL)
```

Arguments

old	Previous version of the queries report.
new	New version of the queries report. The status of each query is determined using this object.
report_title	Character string specifying the title of the report.

Value

A list consisting of a dataframe with every single query of both reports and a column containing the status of the queries (new, solved, miscorrected or pending) compared to the previous query report. There is also a summary of the total number of queries per category in addition to this dataframe.

Examples

```
# Example of a query
data_old <- rd_query(covican,
                     variables = "copd",
                     expression = "is.na(x)",
                     event = "baseline_visit_arm_1")
data_new <- rbind(data_old$queries[1:5,], c("100-20",rep("abc",8)))

# Control of queries
check <- check_queries(old = data_old$queries,
                       new = data_new)
```

covican *Subset of the COVICAN's database*

Description

A random sample of the COVICAN study. An international, multicentre cohort study of cancer patients with COVID-19 to describe the epidemiology, risk factors, and clinical outcomes of co-infections and superinfections in onco-hematological patients with COVID-19.

Usage

```
data(covican)
```

Format

A data frame with 342 rows and 56 columns

record_id: Identifier of each record. This information does not match the real data.

redcap_event_name: Auto-generated name of the events

redcap_data_access_group: Auto-generated name of each center. This information does not match the real data.

inc_1: Inclusion criteria of 'Patients older than 18 years' (0 = No ; 1 = Yes)

inc_2: Inclusion criteria of 'Cancer patients' (0 = No ; 1 = Yes)

inc_3: Inclusion criteria of 'Diagnosed of COVID-19' (0 = No ; 1 = Yes)

exc_1: Exclusion criteria of 'Solid tumour remission >1 year' (0 = No ; 1 = Yes)

screening_fail_crit: Indicator of non-compliance with inclusion and exclusion criteria (0 = compliance ; 1 = non-compliance)

d_birth: Date of birth (y-m-d). This date does not correspond to the original.

d_admission: Date of first visit (y-m-d). This date does not correspond to the original.

age: Age in years

dm: Indicator of diabetes (0 = No ; 1 = Yes)

type_dm: Type of diabetes (1 = No complications ; 2 = End-organ diabetes-related disease)

copd: Indicator of chronic obstructive pulmonary disease (0 = No ; 1 = Yes)

fio2: Fraction of inspired oxygen in percentage

available_analytics: Indicator of blood test available (0 = No ; 1 = Yes)

potassium: Potassium in mmol/L

resp_rate: Respiratory rate in bpm

leuk_lymph: Indicator of leukemia or lymphoma (0 = No ; 1 = Yes)

acute_leuk: Indicator of acute leukemia (0 = No ; 1 = Yes)

type_underlying_disease[... :] Checkbox with the type of underlying disease (0 = Haematological cancer ; 1 = Solid tumour)

underlying_disease_hemato[... :] Checkbox with the type of underlying disease (1 = Acute myeloid leukemia ; 2 = Myelodysplastic syndrome ; 3 = Chronic myeloid leukaemia ; 4 = Acute lymphoblastic leukaemia ; 5 = Hodgkin lymphoma ; 6 = Non Hodgkin lymphoma ; 7 = Multiple myeloma ; 8 = Myelofibrosis ; 9 = Aplastic anaemia ; 10 = Chronic lymphocytic leukaemia ; 11 = Amyloidosis ; 12 = Other)

urine_culture: Indicator of urine culture: (0 = Not done ; 1 = Done)

[... .factor:] Labels of the different variables

Note

List containing three dataframes: the first one with the data, the second one with the dictionary ('codebook') of the REDCap project and the final one with the instrument-event mappings of the REDCap project.

References

Gudiol, C., Durà-Miralles, X., Aguilar-Company, J., Hernández-Jiménez, P., Martínez-Cutillas, M., Fernandez-Avilés, F., Machado, M., Vázquez, L., Martín-Dávila, P., de Castro, N., Abdala, E., Sorli, L., Andermann, T. M., Márquez-Gómez, I., Morales, H., Gabilán, F., Ayaz, C. M., Kayaaslan, B., Aguilar-Guisado, M., Herrera, F. Royo-Cebrecos C, Peghin M, González-Rico C, Goikoetxea J, Salgueira S, Silva-Pinto A, Gutiérrez-Gutiérrez B, Cuellar S, Haidar G, Maluquer C, Marin M, Pallarès N, Carratalà J. (2021). Co-infections and superinfections complicating COVID-19 in cancer patients: A multicentre, international study. The Journal of infection, 83(3), 306–313. https://doi.org/10.1016/j.jinf.2021.07.014

`fill_data` *Fill rows with the values in one event*

Description

Function that with one particular variable and event it fills all the rows in the data with the value in that particular event. Auxiliar to rd_rlogic function

Usage

```
fill_data(which_event, which_var, data)
```

Arguments

`which_event`	String with the name of the event
`which_var`	String with the name of the variable
`data`	Dataset containing the REDCap data.

rd_event *Identification of missing event(s)*

Description

When working with a longitudinal REDCap project, the exported data has a structure where each row represents one event per record. However, by default, REDCap does not export events for which there is no information available. This function allows you to identify which record identifiers do not contain information about a particular event.

Usage

```
rd_event(
  ...,
  data = NULL,
  dic = NULL,
  event,
  filter = NA,
  query_name = NA,
  addTo = NA,
  report_title = NA,
  report_zeros = FALSE,
  link = list()
)
```

Arguments

...	List containing the data, the dictionary and the event (if required). It may be the output of the 'redcap_data' function.
data	Data frame containing the data read from REDCap. If the list is given, this argument is not required.
dic	Data frame containing the dictionary read from REDCap. If the list is given, this argument is not required.
event	Character vector with the name of the REDCap event(s) to be analyzed.
filter	A filter to be applied to the dataset. This argument can be used to identify the missing events on a subset of the dataset.
query_name	Description of the query. It can be defined as the same one for all the variables, or you can define a different one for each variable. By default, the function defines it as 'The event [event] is missing' for each event'.
addTo	Data frame corresponding to a previous query data frame to which you can add the new query data frame. By default, the function always generates a new data frame without taking into account previous reports.
report_title	Character string specifying the title of the report.
report_zeros	Logical. If 'TRUE', the function returns a report containing variables with zero queries.

link	List containing project information used to create a web link to each missing event.

Value

A list with a data frame of 9 columns (10 columns, if the link argument is specified) meant to help the user identify each missing event and a table with the total number of missing events per event analyzed.

Examples

```
example <- rd_event(covican,
                    event = "follow_up_visit_da_arm_1")
example
```

---

rd_export *Exporting the dataset of the queries*

Description

This function allows you to export the previously identified queries to an xlsx file.

Usage

```
rd_export(
  ...,
  queries = NULL,
  column = NULL,
  sheet_name = NULL,
  path = NULL,
  password = NULL
)
```

Arguments

...	List containing the data frame of queries. It may be the output of the 'rd_query' or 'rd_event' functions.
queries	Data frame containing the identified queries. If the list is given, this argument is not required.
column	Character element specifying the column containing the link of each query.
sheet_name	Character element specifying the sheet name of the resulting xlsx file.
path	Character element specifying the file path to save the xlsx file. If 'NULL', the file is created in the current working directory.
password	String with the password to protect the worksheet and prevent others from making changes.

Value

An xlsx file containing all the queries and, if available, hyperlinks to each one of them.

`rd_insert_na` *Insert missing using a filter*

Description

Function that allows you to manually input a missing to some variables (‘vars‘) when some filters (‘filter‘) are satisfied. Useful for checkboxes without a gatekeeper question in the branching logic. Take in account that the variable will be transformed only in the events where both the variable and the filter evaluation are present, so they need to have at least one event in common.

Usage

```
rd_insert_na(..., data = NULL, dic = NULL, event_form = NULL, vars, filter)
```

Arguments

`...`	List containing the data and the dictionary and the event if it’s needed. Can be the output of the function ‘redcap_data‘.
`data`	Data frame containing data from REDCap. If the list is specified this argument is not needed.
`dic`	Data frame containing the dictionary read from REDCap. If the list is specified this argument is not needed.
`event_form`	Data frame containing the correspondence of each event with each form. If the list is specified this argument is not needed.
`vars`	Character vector containing the names of those variables to transform.
`filter`	Character vector containing the logic to be directly evaluated. When each logic is TRUE the corresponding variable specified in ‘vars‘ will be put to missing.

Value

transformed data with the specified variables converted.

Examples

```
table(is.na(covican$data$potassium))
data <- rd_insert_na(covican,
            vars = "potassium",
            filter = "age < 65")
table(data$potassium)
```

rd_query *Identification of queries*

Description

This function allows you to identify queries by the use a specific expression. It can be used to identify missing values or to identify values that are outside the lower and upper limits of a variable.

Usage

```
rd_query(
  ...,
  variables = NA,
  expression = NA,
  negate = FALSE,
  event = NA,
  filter = NA,
  addTo = NA,
  variables_names = NA,
  query_name = NA,
  instrument = NA,
  report_title = NA,
  report_zeros = FALSE,
  by_dag = FALSE,
  link = list(),
  data = NULL,
  dic = NULL,
  event_form = NULL
)
```

Arguments

...	List containing the data, the dictionary and the event (if required). It may be the output of the 'redcap_data' function.
variables	Character vector of the names of the database variables to be checked.
expression	Character vector of expressions to be applied to the selected variables.
negate	Logical value which indicates whether the defined expression should be negated. Defaults to 'FALSE'.
event	The name of the REDCap event to be analyzed. If there are events in your REDCap project, you should use this argument in order to name the event to which the defined variables belong.
filter	A filter to be applied to the dataset. For example, the branching logic of a determined variable can be applied using this argument.
addTo	Data frame corresponding to a previous query data frame to which you can add the new query data frame. By default, the function always generates a new data frame without taking into account previous reports.

variables_names
: Character vector containing the description of each selected variable. By default, the function automatically takes the description of each variable from the dictionary of the REDCap project.

query_name
: Description of the query. It can be defined as the same one for all the variables, or you can define a different one for each variable. By default, the function defines it as 'The value is [value] and it should not be [expression]''.

instrument
: REDCap's instrument to which the variables belong. It can be defined as the same one for all the variables, or you can define a different one for each variable. By default, the function automatically selects the corresponding instrument of each variable from the dictionary of the REDCap project.

report_title
: Character string specifying the title of the report.

report_zeros
: Logical. If 'TRUE', the function returns a report containing variables with zero queries.

by_dag
: Logical. If 'TRUE', both elements of the output are grouped by the data access groups (DAGs) of the REDCap project.

link
: List containing project information used to create a web link to each query.

data
: Data frame containing the data read from REDCap. If the list is given, this argument is not required.

dic
: Data frame containing the dictionary read from REDCap. If the list is given, this argument is not required.

event_form
: Data frame containing the correspondence of each event with each form. If the list is specified this argument is not necessary.

Value

A list with a data frame of 9 columns (10 columns, if the link argument is specified) meant to help the user identify each query and a table with the total number of queries per variable.

Examples

```
# Missing values
example <- rd_query(covican,
                    variables = c("copd", "age"),
                    expression = c("is.na(x)", "x %in% NA"),
                    event = "baseline_visit_arm_1")
example

# Expression
example <- rd_query(covican,
                    variables="age",
                    expression="x>20",
                    event="baseline_visit_arm_1")
example

# Using the filter argument
example <- rd_query(covican,
                    variables = "potassium",
```

```
                    expression = "is.na(x)",
                    event = "baseline_visit_arm_1",
                    filter = "available_analytics=='1'")
example
```

---

rd_rlogic *REDCap logic into R logic*

Description

This function allows you to transcribe REDCap logic to R logic. WARNING: If the REDCap logic involves some smart-variables this function will not be able to transform it.

Usage

```
rd_rlogic(..., data = NULL, dic = NULL, event_form = NULL, logic, var)
```

Arguments

...	List containing the data and the dictionary and the event if it's needed. Can be the output of the function 'redcap_data'.
data	Data frame containing data from REDCap. If the list is specified this argument is not needed.
dic	Data frame containing the dictionary read from REDCap. If the list is specified this argument is not needed.
event_form	Data frame containing the correspondence of each event with each form. If the list is specified this argument is not needed.
logic	String containing a logic in REDCap format.
var	string with the name of the variable that contains the logic.

Value

List containing the logic in R format and its evaluation.

Examples

```
rd_rlogic(covican,
          logic = "if([exc_1]='1' or [inc_1]='0' or [inc_2]='0' or [inc_3]='0',1,0)",
          var = "screening_fail_crit")
```

`rd_transform` *Transformation of the raw data*

Description

Function that transforms the raw data from REDCap read by the function 'redcap_data'. It returns the transformed data and dictionary along with the summary of the results of each step.

Usage

```
rd_transform(
  ...,
  data = NULL,
  dic = NULL,
  event_form = NULL,
  checkbox_labels = c("No", "Yes"),
  checkbox_na = FALSE,
  exclude_recalc = NULL,
  exclude_to_factor = NULL,
  delete_vars = NULL,
  delete_pattern = c("_complete", "_timestamp"),
  final_format = "raw",
  which_event = NULL,
  which_form = NULL,
  wide = NULL
)
```

Arguments

`...`	Output of the function 'redcap_data', that is a list containing the data frames of the data, he dictionary and the event_form (if it's needed) of the REDCap project.
`data`	Data frame containing the data read from REDCap. If the list is specified this argument is not necessary.
`dic`	Data frame containing the dictionary read from REDCap. If the list is specified this argument is not necessary.
`event_form`	Data frame containing the correspondence of each event with each form. If the list is specified this argument is not necessary.
`checkbox_labels`	Character vector with the names that will have the two options of every checkbox variable. Default is 'c('No', 'Yes')'.
`checkbox_na`	Logical indicating if values of checkboxes that have a branching logic have to set to missing only when the branching logic is missing (if set to false) or also when the branching logic isn't satisfied (if set to true). The default is false.

exclude_recalc Character vector with the names of the variables that do not have to be recalculated. Might be useful for projects were there are some calculated fields that have a time consuming recalculation.

exclude_to_factor Character vector with the names of the variables that do not have to be transformed to factors.

delete_vars Character vector specifying the variables to exclude.

delete_pattern Character vector specifying the regex pattern that will contain the variables to exclude. By default, variables ending up with '_complete' and '_timestamp' will be removed.

final_format Character string indicating the final arrangement format of the data that the function will return. Choose one of 'raw', 'by_event' or 'by_form'. 'raw' (default) will return the transformed data with the original structure. 'by_event' will return the transformed data as a nested data frame by event. 'by_form' will return the transformed data as a nested data frame by form.

which_event Character string indicating if only one event has to be returned if the final format selected is 'by_event'.

which_form Character string indicating if only one form has to be returned if the final format selected is 'by_form'.

wide Logical indicating if the data split by form (if selected) has to be in a wide format or in a long one.

Value

List with the transformed dataset, dictionary, event_form and the results

Examples

```
rd_transform(covican)

# For customization of checkbox labels
rd_transform(covican,
             checkbox_labels = c("Not present", "Present"))
```

---

recalculate *Recalculate REDCap calculated fields*

Description

Function that recalculates every calculated field if the logic can be transcribed to R. Recall that calculated fields with smart-variables in the logic or variables in other events cannot be transcribed.

The function will return the dataset and dictionary with the added recalculated variables (the name of the calculated field + '_recalc') along with a table that shows the summary of the results.

Usage

```
recalculate(data, dic, event_form = NULL, exclude_recalc = NULL)
```

Arguments

data	Data frame containing data from REDCap.
dic	Data frame containing the dictionary read from REDCap.
event_form	Data frame containing the correspondence of each event with each form.
exclude_recalc	Character vector with the names of the variables that do not have to be recalculated. Might be useful for projects were there are some calculated fields that have a time consuming recalculation.

---

redcap_data *Read REDCap data*

Description

This function allows users to read datasets from a REDCap project into R for analysis, either via export of the data or via an API connection.

The REDCap API is an interface that allows communication with REDCap and server without going through the interactive REDCap interface.

Usage

```
redcap_data(
  data_path = NA,
  dic_path = NA,
  event_path = NA,
  uri = NA,
  token = NA,
  filter_field = NA
)
```

Arguments

data_path	Character string with the pathname of the R file to read the dataset from.
dic_path	Character string with the pathname of the dictionary.
event_path	Character string with the pathname of the file containing the correspondence between each event and each form (it can be downloaded through the 'Designate Instruments for My Events' tab inside the 'Project Setup' section of REDCap)
uri	The URI (Uniform Resource Identification) of the REDCap project.
token	Character vector with the generated token.
filter_field	Character vector with the fields of the REDCap project desired to import into R (API connection only)<.

Value

List containing the dataset and the dictionary of the REDCap project. If the event_path is specified, it will also contain a third element with the correspondence of the events & forms of the project.

Note

If you will give further use to the package, we advise you to use the argument 'dic_path' to read your dictionary, as all other functions need it in order to run properly.

To read exported data, you must first use REDCap's 'Export Data' function and select the 'R Statistical Software' format. It will then generate a CSV file with all the observations and an R file with the necessary code to complete each variable's information.

Examples

```
## Not run:
# Exported files from REDCap

dataset <- redcap_data(data_path = "C:/Users/username/example.r",
                       dic_path = "C:/Users/username/example_dictionary.csv",
                       event_path = "C:/Users/username/events.csv")

# API connection

dataset_api <- redcap_data(uri = "https://redcap.idibell.cat/api/",
                       token = "55E5C3D1E83213ADA2182A4BFDEA") # This token is fictitious


## End(Not run)
```

---

split_event *Creation of a data frame with variables of all the forms of a specified event*

Description

Function that given the data, the dictionary and the mapping between forms and events it creates a nested dataset containing all the datasets filtered by each event and containing only the variables found in the event. It can be chosen to return only the data from the specified event.

Usage

```
split_event(data, dic, event_form, which = NULL)
```

Arguments

data	Data frame containing data from REDCap.
dic	Data frame containing the dictionary read from REDCap.
event_form	Data frame containing the correspondence of each event with each form.
which	Specify an event if only data for the desired event is wanted.

---

split_form *Creation of a data frame with variables of a specified form*

Description

Function that given the data, the dictionary and the mapping between forms and events it creates a nested dataset containing all the datasets having only the variables in each form. It can be chosen to return only the data from the specified form

Usage

```
split_form(data, dic, event_form = NULL, which = NULL, wide = FALSE)
```

Arguments

data	Data frame containing data from REDCap.
dic	Data frame containing the dictionary read from REDCap.
event_form	Data frame containing the correspondence of each event with each form.
which	Specify a form if only data for the desired form is wanted.
wide	If the dataset needs to be in a wide format or not (long).

---

to_factor *Convert variables to factors*

Description

Function that converts every variable except those specified to factor.

Usage

```
to_factor(data, dic, exclude = NULL)
```

Arguments

data	Dataset containing the REDCap data.
dic	Dataset containing the REDCap dictionary.
exclude	Character vector containing the names of those variables that will not be converted to factors. If 'NULL', all variables will be converted.

`transform_checkboxes` *Transformation of checkboxes in case of having a branching logic*

Description

Inspects all the checkboxes of the study and looks if there is a branching logic. If there is one, when the logic of the branching logic is missing it directly inputs a missing to the checkbox. If checkbox_na is TRUE additionally it will put a missing when the branching logic isn't satisfied and not only when the logic is missing. If a branching logic cannot be found or the logic cannot be transcribed because of the presence of some smart variables, the variable is added in the list of the reviewable ones that will be printed.

The function will return the dataset with the transformed checkboxes along with a table that shows a summary of the results.

Usage

```
transform_checkboxes(data, dic, event_form = NULL, checkbox_na = FALSE)
```

Arguments

`data`	Data frame containing data from REDCap.
`dic`	Data frame containing the dictionary read from REDCap.
`event_form`	Data frame containing the correspondence of each event with each form.
`checkbox_na`	Logical indicating if values of checkboxes that have a branching logic have to set to missing only when the branching logic is missing (if set to false) or also when the branching logic isn't satisfied (if set to true). The default is false.

Index

∗ **datasets**
covican, 4

check_queries, 3
checkbox_names, 2
covican, 4

fill_data, 5

rd_event, 6
rd_export, 7
rd_insert_na, 8
rd_query, 9
rd_rlogic, 11
rd_transform, 12
recalculate, 13
redcap_data, 14

split_event, 15
split_form, 16

to_factor, 16
transform_checkboxes, 17