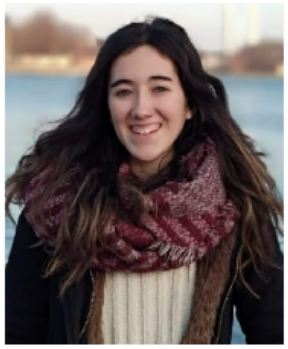
Managing REDCap Data: The R package REDCapDM

Unitat de Bioestadística de l'IDIBELL

L'Hospitalet de LLobregat, Febrero 2023

Nosaltres

UBiDi



IDI BELL
Institut d'Investigació Biomèdica de Bellvitge

REDCap

Què és?

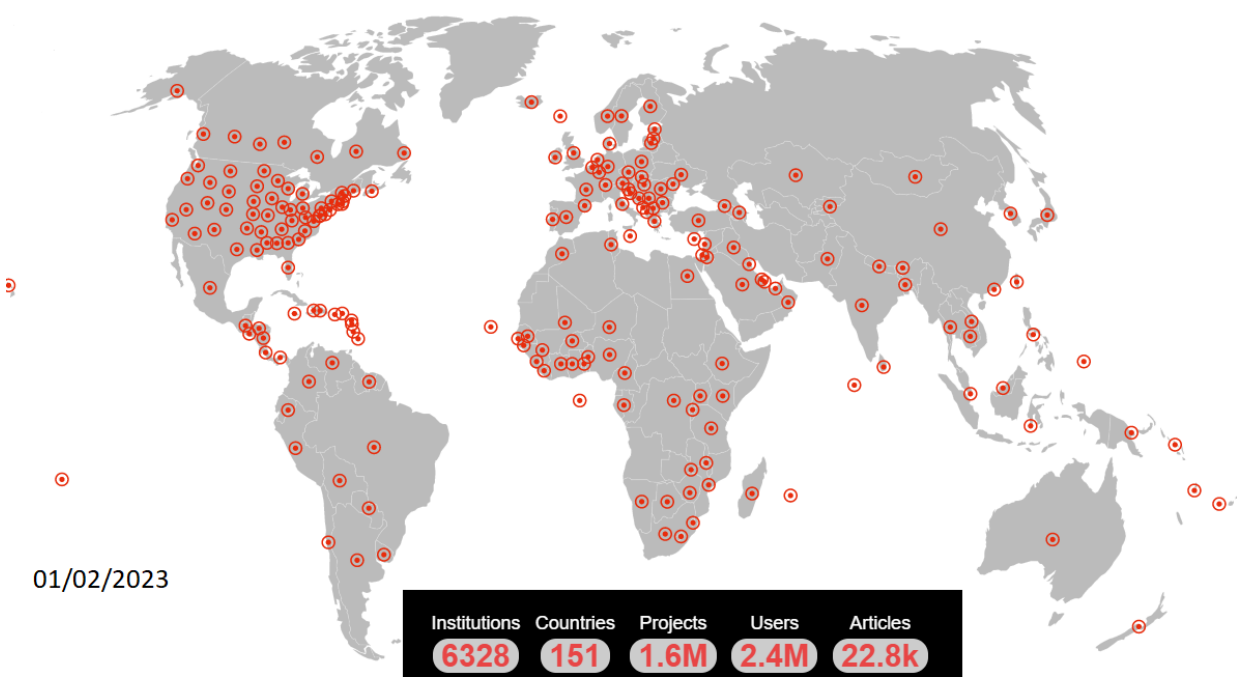
Aplicació web per l'entrada de dades en estudis de recerca.

- **Vanderbilt University** (2004).
- Acord de llicència propietat de la universitat de Vanderbilt ús no comercial.

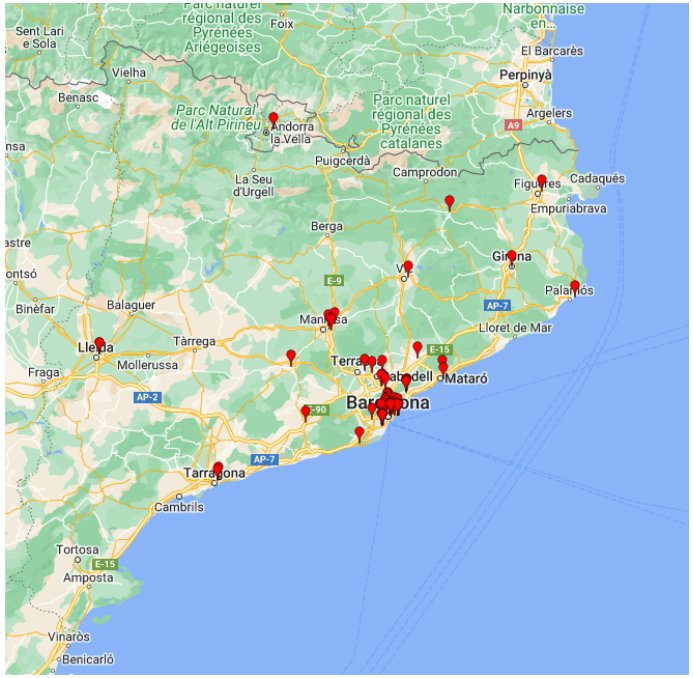
Característiques principals

- Web-based
- Multi-Accés
- Traçabilitat
- Exportació de dades
- Seguretat
- Actualitzacions i millores constants

Situació actual



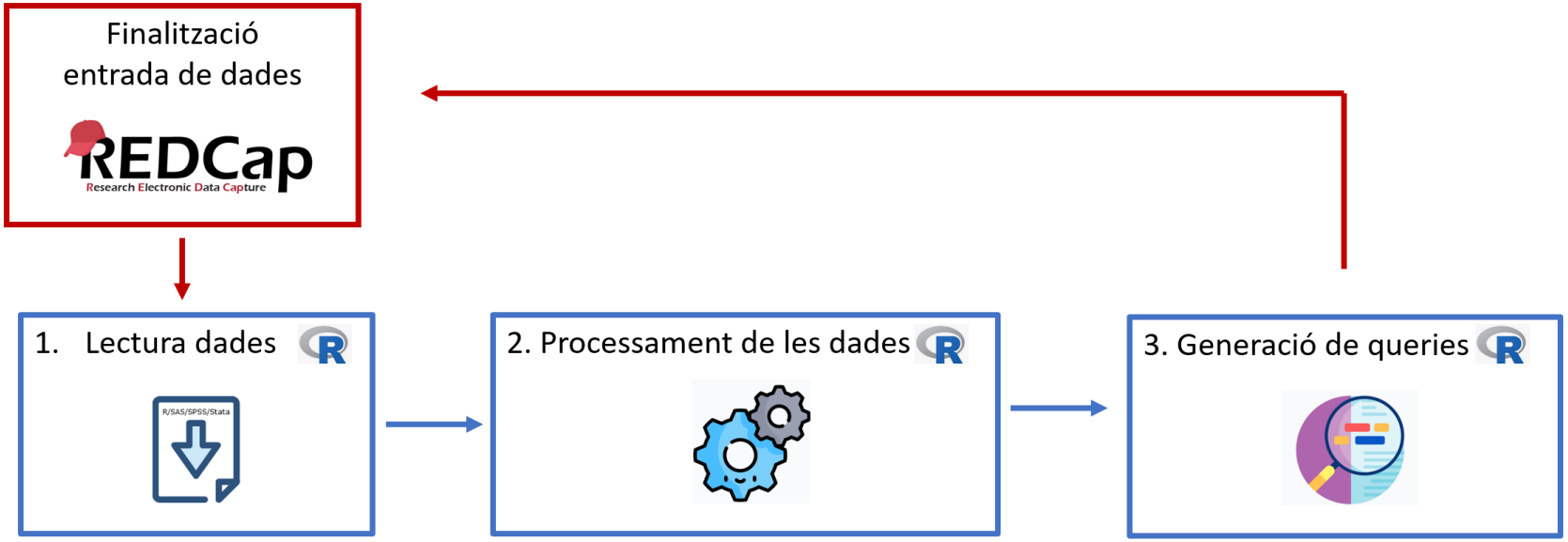
01/02/2023



Paquet REDCapDM

Motivació

Agilitzar i automatitzar els processos que se situen entre la finalització d'entrada de dades i l'obtenció d'una base de dades llesta per l'anàlisi estadística en R:



Motivació

1. Lectura de les dades en R

Automatitzar la lectura de dades agilitza obtenir objectes de l'entorn d'R.

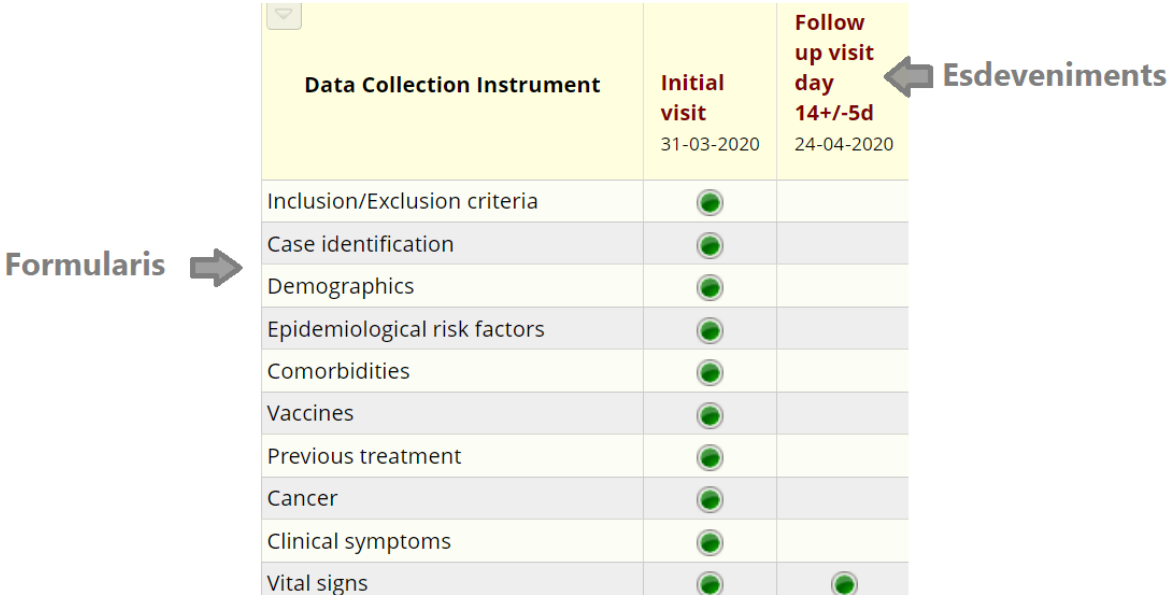
Motivació

2. Processament de les dades REDCap en R

Algunes característiques de les dades proporcionades per REDCap requereixen processament en R.

A continuació veurem les principals particularitats de REDCap a tenir en compte:

- **Estructura dels formularis/esdeveniments**



Data Collection Instrument	Initial visit 31-03-2020	Follow up visit day 14+/-5d 24-04-2020
Inclusion/Exclusion criteria	●	
Case identification	●	
Demographics	●	
Epidemiological risk factors	●	
Comorbidities	●	
Vaccines	●	
Previous treatment	●	
Cancer	●	
Clinical symptoms	●	
Vital signs	●	●

Motivació

2. Processament de les dades REDCap en R

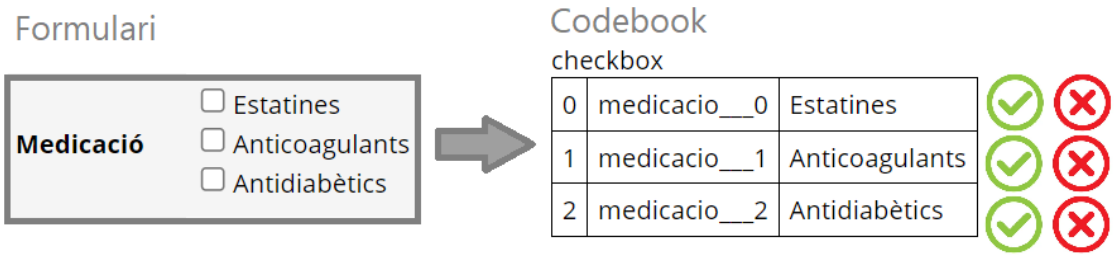
- Codebook

	#	Variable / Field Name	Field Label *Field Note*	Field Attributes (Field Type, Validation, Choices, Calculations, etc.)
Instrument: **Inclusion/Exclusion criteria** (inclusionexclusion_criteria)				Expand
Instrument: **Case identification** (case_identification)				Expand
Instrument: **Demographics** (demographics)				Collapse
	21	[d_ingreso]	Date of first visit	text (date_dmy)
	22	[hospital_admission]	Hospital admission	radio 0 No 1 Yes

Motivació

2. Processament de les dades REDCap en R

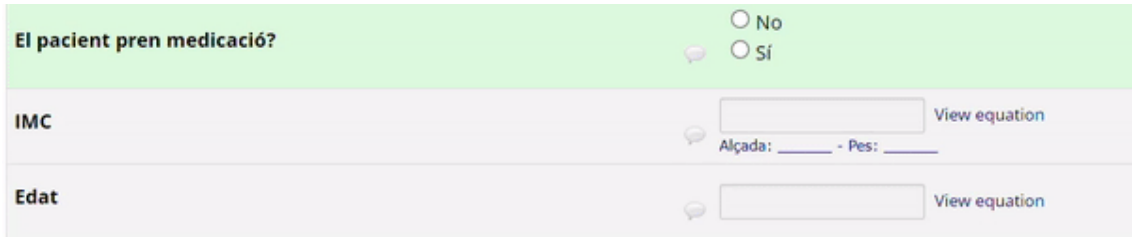
- **Checkbox**



Motivació

2. Processament de les dades REDCap en R

- Branching logics



Motivació

2. Processament de les dades REDCap en R

- Camps autocalculats

calc
Calculation: datediff([d_nacimiento],[d_visita_basal],"y","dmy")

calc
Calculation: datediff([inclusion][d_nacimiento],[d_seguimient0],"y","dmy")

Automatitzar el processament de les dades proporcionades pel REDCap agilitza l'obtenció d'una base de dades llesta per les anàlisis en el software R.

Motivació

3.Generació de queries i el seu seguiment en 

Moltes vegades és necessari realitzar preguntes sobre la nostra base de dades amb l'objectiu de validar-la i fer un seguiment de la revisió:

- Dates errònies
- Valors fora dels rangs establerts en REDCap
- Dades Missing

Automatitzar la generació de queries i el seu seguiment pot comportar reduir el seu temps elevat de programació

Funcionament REDCapDM

CRAN

REDCapDM: 'REDCap' Data Management

Access and manage 'REDCap' data. 'REDCap' (Research Electronic Data CAPture; <https://projectredcap.org>) is a web application for building and managing online surveys and databases developed at Vanderbilt University. The API allows users to programmatic access data and project meta data (such as the data dictionary) from the web. This package allows us to read 'REDCap' data, exported or using an API connection, identify missing or extreme values, identify missing 'REDCap' events in each observation, do a follow-up of the queries initially identified and it also facilitates the process of data management.

Version:	0.1.1
Depends:	R (≥ 2.10)
Imports:	dplyr, REDCapR, janitor, stringr, magrittr, tidyr, Hmisc, utils, purrr, tidyselect, tibble, rlang
Suggests:	knitr, rmarkdown, kableExtra
Published:	2023-01-30
Author:	João Carmezim [aut, cre], Judith Peñafiel [aut], Pau Satorra [aut], Esther García [aut], Natàlia Pallarés [aut], Cristian Tebé [aut]
Maintainer:	João Carmezim <ubidi at idibell.cat>
BugReports:	https://github.com/ubidi/REDCapDM/issues
License:	MIT + file LICENSE
NeedsCompilation:	no
Materials:	NEWS
CRAN checks:	REDCapDM results

Documentation:

Reference manual:	REDCapDM.pdf
Vignettes:	REDCapDM

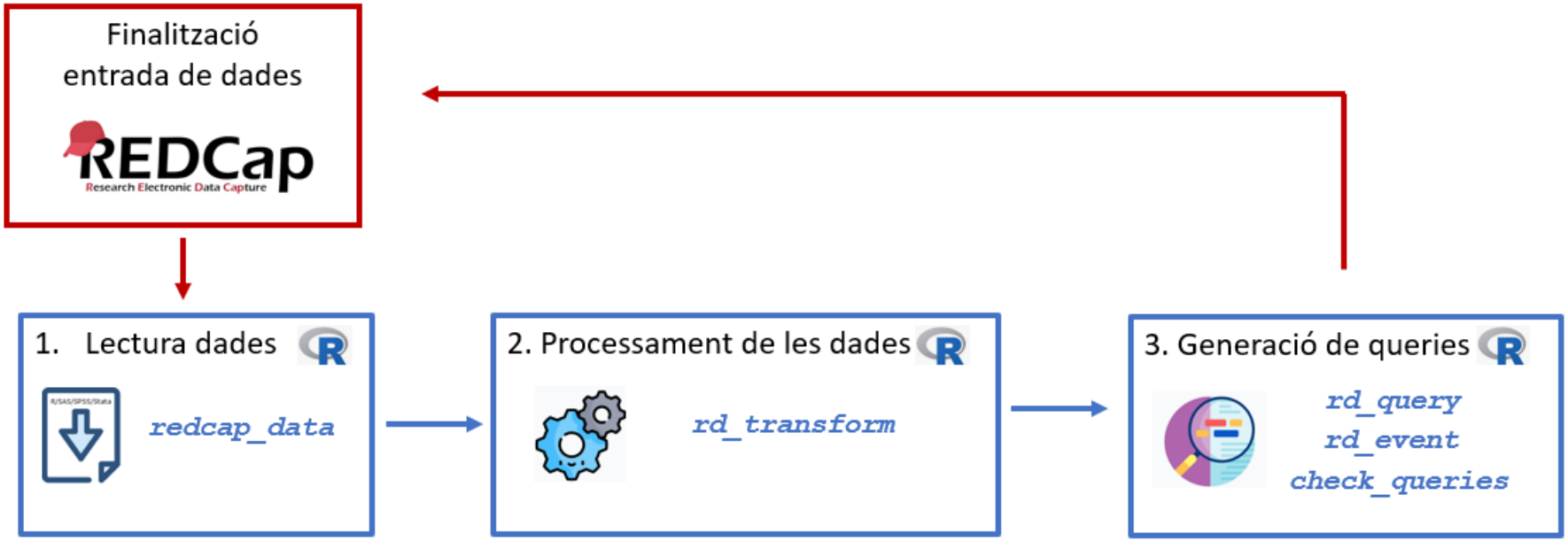
Downloads:

Package source:	REDCapDM_0.1.1.tar.gz
Windows binaries:	r-devel: REDCapDM_0.1.1.zip, r-release: REDCapDM_0.1.1.zip, r-oldrel: REDCapDM_0.1.0.zip
macOS binaries:	r-release (arm64): REDCapDM_0.1.1.tgz, r-oldrel (arm64): REDCapDM_0.1.1.tgz, r-release (x86_64): REDCapDM_0.1.0.tgz, r-oldrel (x86_64): REDCapDM_0.1.0.tgz
Old sources:	REDCapDM archive

Linking:

Please use the canonical form https://CRAN.R-project.org/package=REDCapDM to link to this page.

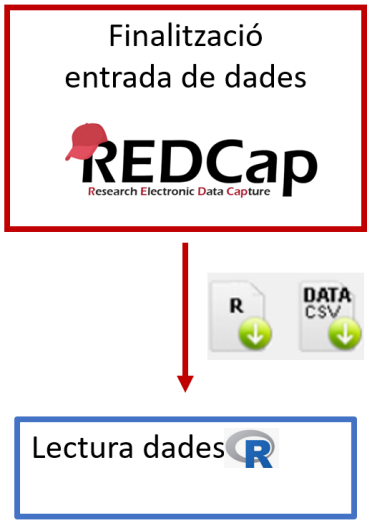
Estructura


Finalització
entrada de dades
REDCap
Research Electronic Data Capture
1. Lectura dades
redcap_data
2. Processament de les dades
rd_transform
3. Generació de queries
rd_query
rd_event
check_queries

1. Lectura de dades

Lectura de dades

(Descarregar del REDCap)



1. Descarregar les dades (csv) i el fitxer R del REDCap a **una mateixa carpeta**

2. Descarregar el diccionari (codebook) i guardar-lo en qualsevol carpeta

3. Utilitzar la funció `redcap_data`:

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

```
str(dataset, max.level = 1)
```

```
List of 2
 $ data      :'data.frame': 100 obs. of  30 variables:
 $ dictionary:'data.frame': 15 obs. of  8 variables:
```

Lectura de dades

(Connexió API)

1. Generar una API token del projecte en el REDCap

2. Utilitzar la funció `redcap_data`:

```
dataset <- redcap_data(uri ="https://redcap.idibell.cat/api/",
                       token = "55E5C3D1E83213ADA2182A4BFDEA")
```

Cal tenir cura al compartir el token amb tercers

Exemple de base de dades

(*COVICAN*)

- El paquet té incorporada una base de dades que prové d'un projecte real de REDCap: *COVICAN*
- *COVICAN* és un estudi de cohorts internacional i multicèntric sobre l'epidemiologia, els factors de risc i els resultats clínics de les coinfeccions i sobreinfeccions en pacients oncohematològics amb COVID-19
- La base de dades incorporada és una mostra aleatòria de l'estudi simplificada llegida amb `redcap_data`

```
str(covican, max.level = 1)
```

```
List of 2
 $ data      :'data.frame': 342 obs. of  56 variables:
 $ dictionary:'data.frame': 21 obs. of  8 variables:
```

Exemple de base de dades

(*COVICAN*)

Les bases de dades de REDCap contenen una fila per pacient-esdeveniment:

record_id	redcap_event_name	d_birth	age	dm	potassium
100-6	baseline_visit_arm_1	1963-10-05	56	0	4.30
100-6	follow_up_visit_da_arm_1				4.50
100-13	baseline_visit_arm_1	1959-10-04	60	0	3.66
100-13	follow_up_visit_da_arm_1				4.10

2. Processament

Processament

- Utilitzarem la base de dades `covican` com a exemple
- Cridarem la funció `rd_transform` amb els següents arguments com a mínim:

```
covican_transformed <- rd_transform(data = covican$data,
                                    dic = covican$dictionary)
```

```
str(covican_transformed, max.level = 1)
```

```
List of 3
 $ data      :'data.frame': 342 obs. of  36 variables:
 $ dictionary:'data.frame': 32 obs. of  8 variables:
 $ results   : chr "1. Recalculating calculated fields and saving them as..."
```

Processament

1. Recalculating calculated fields and saving them as '[field_name]_recalc'

Total calculated fields	Non-transcribed fields	Recalculated different fields
2	0 (0%)	1 (50%)

field_name	Transcribed?	Is equal?
age	Yes	FALSE
screening_fail_crit	Yes	TRUE

2. Transforming checkboxes: changing their values to No/Yes and changing their names to the names of its options.
For checkboxes that have a question door specified in the branching logic, converting some of their values to missing

Table: Checkbox variables advisable to be reviewed

Variables without any branching logic
type_underlying_disease

3. Replacing original variables for their factor version

4. Deleting variables that contain some patterns

Processament

(Estructura per esdeveniment)

- Podem canviar l'estructura de les dades transformades separant-les per cada esdeveniment
- Cal descarregar-se la correspondència entre els esdeveniments i els formularis en el REDCap del projecte:

```
covican_transformed <- rd_transform(data = covican$data,
                                    dic = covican$dictionary,
                                    event_path = "files/COVICAN_instruments.csv",
                                    final_format = "by_event")
```

S'han afegit dos passos extra:

```
5. Erasing variables from forms that are not linked to any event


6. Final arrangment of the data by event
```

Processament

(Estructura per esdeveniment)

Les dades transformades tenen la següent estructura:

```
covican_transformed$data
```

```
# A tibble: 2 × 3
  events                   vars       df
  <chr>                    <list>     <list>
1 baseline_visit_arm_1     <chr [25]> <df [190 × 25]>
2 follow_up_visit_da_arm_1 <chr [8]>  <df [152 × 8]>
```

Processament

(Estructura per formulari)

En lloc d'estructurar les dades transformades per esdeveniment ho podem fer per formulari:

```
covican_transformed <- rd_transform(data = covican$data,
                                    dic = covican$dictionary,
                                    event_path = "files/COVICAN_instruments.csv",
                                    final_format = "by_form")
```

```
covican_transformed$data
```

```
# A tibble: 6 × 4
  form                        events    vars       df
  <chr>                       <list>    <list>     <list>
1 inclusionexclusion_criteria <chr [1]> <chr [11]> <df [190 × 11]>
2 demographics                <chr [1]> <chr [9]>  <df [190 × 9]>
3 comorbidities               <chr [1]> <chr [10]> <df [190 × 10]>
4 vital_signs                 <chr [2]> <chr [7]>  <df [177 × 7]>
5 laboratory_findings         <chr [2]> <chr [7]>  <df [177 × 7]>
6 microbiological_studies     <chr [1]> <chr [6]>  <df [190 × 6]>
```

Processament

(Estructura per formulari)

Si un formulari conté més d'un esdeveniment la base de dades corresponent tindrà una fila per pacient i esdeveniment:

```
covican_transformed$data %>%
  filter(form == "vital_signs") %>%
  unnest(df)
```

record_id	redcap_event_name	fio2	resp_rate
100-13	baseline_visit_arm_1	21	16
100-13	follow_up_visit_da_arm_1	21	
101-16	baseline_visit_arm_1	35	20
101-16	follow_up_visit_da_arm_1	100	

3. Queries

Queries

- Usaremos la base de datos `covican_transformed`, resultante del procesamiento de la base de datos inicial.

- Miraremos:
  - Missings y valores inconsistentes con la función `rd_query`
  - Eventos missing con la función `rd_event`
  - Seguimiento de queries con la función `check_queries`

Ejemplo de output

Una lista compuesta por dos elementos ↴

- Data frame con todas las queries identificadas:

Identifier	DAG	Event	Instrument	Field	Repetition	Description	Query	Code
102-113	Hospital 24	Baseline visit	Demographics	age	.	Age	The value is NA and it should not be missing	102-113-1

- Resumen del número total de queries por variable:

Variables	Description	Total
age	Age	1

Generar queries

(Identificación de missings)

Información necesaria para un uso correcto de la función:

```
example <- rd_query(variables = "d_birth",
                    expression = "%in%NA",
                    event = "baseline_visit_arm_1",
                    dic = covican_transformed$dictionary,
                    data = covican_transformed$data)
```

```
List of 2
 $ queries:'data.frame':    5 obs. of  9 variables:
 $ results: 'knitr_kable' chr [1:5] "Table: Report of queries" "-" "-" "-" ...
  ..- attr(*, "format")= chr "pipe"
```

```
example$results
```

Variables	Description	Total
d_birth	Date of birth	5

Generar queries

(Missings con branching logic)

¿Qué pasa cuando intentamos identificar missings en una variable que presenta un branching logic?

```
example <- rd_query(variables = "potassium",
                    expression = "%in%NA",
                    event = "baseline_visit_arm_1",
                    dic = covican_transformed$dictionary,
                    data = covican_transformed$data)
```

```
Warning: Some of the variables that were checked for missings present a branching logic.
Check the results tab of output for more details (...$results).
```

```
example$results
```

Variables	Description	Total	Branching.logic
potassium	Potassium	31	[available_analytics]='1'

Generar queries

(Missings con branching logic - filter)

Podemos usar el argumento `filter` para asegurarnos que el branching logic se cumple:

```
example <- rd_query(variables = "potassium",
                    expression = "%in%NA",
                    event = "baseline_visit_arm_1",
                    filter = "available_analytics=='Yes'",
                    dic = covican_transformed$dictionary,
                    data = covican_transformed$data)
```

Resumen sin filtro:

Variables	Description	Total	Branching.logic
potassium	Potassium	31	[available_analytics]='1'

Resumen con filtro:

Variables	Description	Total	Branching.logic
potassium	Potassium	21	[available_analytics]='1'

Generar queries

(Consultas específicas)

Si, en vez de missings, lo que queremos hacer es identificar valores inconsistentes:

```
example <- rd_query(variables = "age",
                    expression = "<35 | >80",
                    event = "baseline_visit_arm_1",
                    dic = covican_transformed$dictionary,
                    data = covican_transformed$data)
```

```
example$results
```

Variables	Description	Total
age	Age	26

Generar queries

(Versatilidad de la función)

Hasta el momento, solo hemos visto ejemplos de una consulta en concreto para una variable. Sin embargo, podemos hacer múltiplas consultas en simultáneo:

```
example <- rd_query(variables = c("age", "resp_rate", "dm"),
                    expression = c("<30", ">50", "%in%NA"),
                    event = "baseline_visit_arm_1",
                    dic = covican_transformed$dictionary,
                    data = covican_transformed$data)
```

También podemos aplicar la misma consulta a varias variables:

```
example <- rd_query(variables = c("age", "resp_rate", "dm"),
                    expression = c("%in%NA"),
                    event = "baseline_visit_arm_1",
                    dic = covican_transformed$dictionary,
                    data = covican_transformed$data)
```

Generar queries

(Argumento - addTo)

El argumento `addTo` nos permite acumular todas las queries que vayamos identificando en un mismo objeto de R:

```
example2 <- rd_query(variables = "d_birth",
                     expression = "%in%NA",
                     event = "baseline_visit_arm_1",
                     dic = covican_transformed$dictionary,
                     data = covican_transformed$data,
                     addTo = example)
```

Resumen anterior:

Variables	Description	Total
age	Age	26

Resumen actualizado:

Variables	Description	Total
age	Age	26
d_birth	Date of birth	5

Generar queries

(Identificación de eventos missing)

Cuando trabajamos con un proyecto de REDCap que tiene eventos, deberemos tener en cuenta que REDCap no exporta las filas correspondientes a eventos que no tienen información.

Así que, para saber el número de eventos que no se han exportado, usamos la siguiente función:

```
example <- rd_event(event = "follow_up_visit_da_arm_1",
                    dic = covican_transformed$dictionary,
                    data = covican_transformed$data)
```

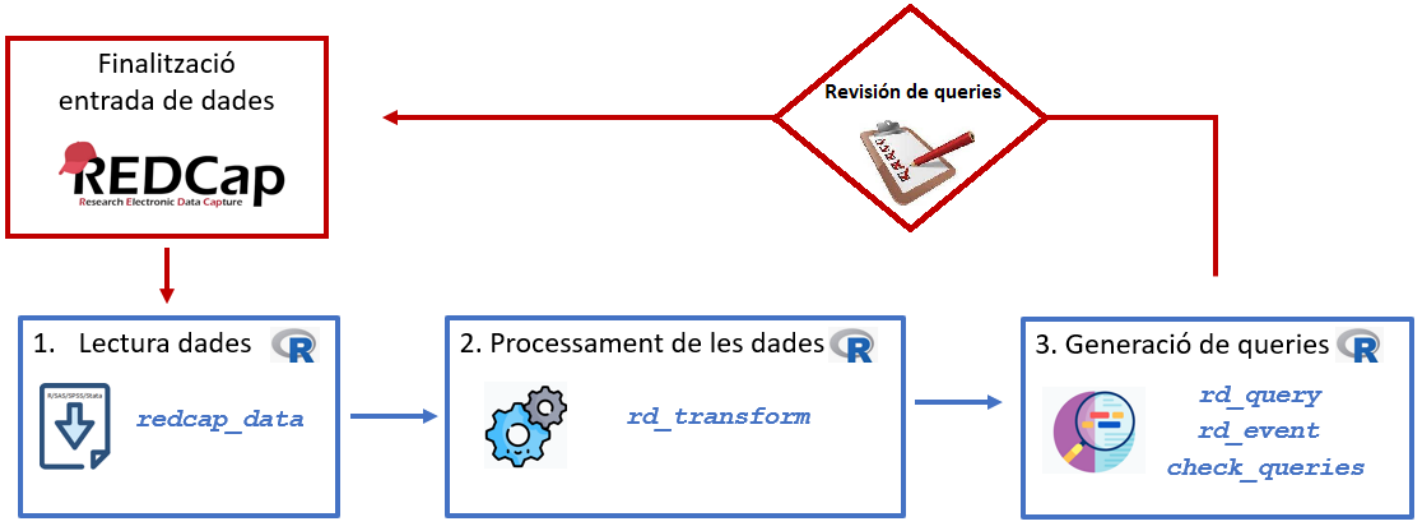
Ahora el resumen corresponderá al número de missings según evento:

```
example$results
```

Events	Description	Total
follow_up_visit_da_arm_1	Follow up visit day 14+/-5d	38

Seguimiento de queries

Finalizado el proceso de identificación, el procedimiento típico sería revisar todas las queries y, una vez revisadas, volver a ejecutar todas las consultas generando así un nuevo dataset de queries.



Seguimiento de queries

Para realizar un control sobre las queries, podemos usar la función `check_queries`:

```
check <- check_queries(old = example$queries,
                       new = new_example$queries)
```

La función compara el dataset de queries antiguo con el recién creado y clasifica cada query en nueva, sin modificar, o modificada.

Atención: una query que se clasifique como “Modificada” no significa que esté resuelta, ya que su modificación puede originar una nueva query.

Seguimiento de queries

(Ejemplo de output)

Sigue siendo una lista con los mismos 2 elementos, pero ↴

- Hay una columna adicional en el dataset de queries con la clasificación de cada query:

Identifier	DAG	Event	Instrument	Field	Repetition	Description	Query	Code	Modification
100-58	Hospital 11	Baseline visit	Comorbidities	copd	.	Chronic obstructive pulmonary disease	The value is NA and it should not be missing	100-58-1	Unmodified

  - El resumen ahora contempla el número de queries según categoría:

State	Total
Unmodified	7
Modified	4
New	1

Limitaciones y futuro

Limitaciones y futuro

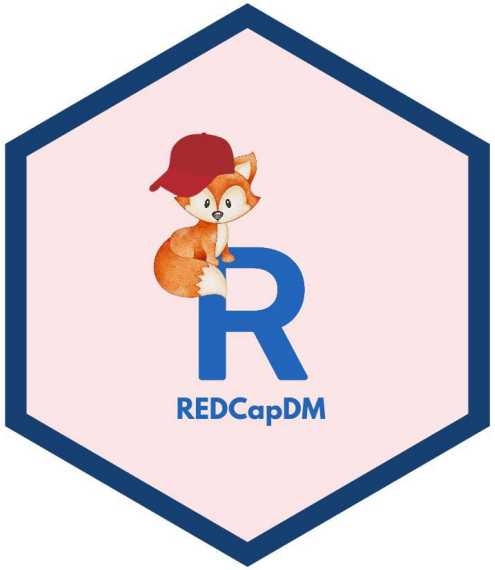
- No fue testado en todas las posibles estructuras de un proyecto de REDCap.
- La última versión del paquete fue testada en la versión 12.4.17 de REDCap.
- El procesamiento de los datos puede no ser apto para manejar lógica compleja de REDCap que incluya smart-variables específicas.
- Intentar conectar el sistema de queries generado en R al sistema existente en REDCap.
- Desarrollo de una aplicación Shiny para complementar el paquete.

Conclusión

Conclusión

El paquete `REDCapDM`:

- Proporciona un conjunto de funciones útiles para importar, organizar y comprobar la calidad de los datos de REDCap.
- Facilita el flujo de trabajo de la gestión de datos de un estudio y ayuda a garantizar datos fiables y de alta calidad que estén listos para análisis.
- Llena un vacío en las herramientas disponibles para la gestión de datos REDCap.



Referencias

- Carmezim J, Peñafiel J, Satorra P, García E, Pallarés N, Tebé C (2022). REDCapDM: 'REDCap' Data Management. R package - version 0.1.0, https://CRAN.R-project.org/package=REDCapDM

- R Core Team (2022). R: A language and environment for statistical computing. R Foundation for Statistical Computing, - Vienna, Austria. URL https://www.R-project.org/.
- Wickham H, François R, Henry L, Müller K (2022). dplyr: A Grammar of Data Manipulation. R package version 1.0.10, - https://CRAN.R-project.org/package=dplyr.
- Will Beasley (2022). REDCapR: Interaction Between R and REDCap. R package version 1.1.0, - https://CRAN.R-project.org/package=REDCapR
- Wickham H (2022). stringr: Simple, Consistent Wrappers for Common String Operations. https://stringr.tidyverse.org, https://github.com/tidyverse/stringr.
- Wickham H, Henry L (2023). purrr: Functional Programming Tools. https://purrr.tidyverse.org/, https://github.com/tidyverse/purrr.