

Análisis de datos como desarrollo

Desde abrir los datos hasta la generación de resultados

Carrasco, D., PhD & Miranda, D., PhD

Centro de Medición MIDE UC

Castillo, C., Mg.

Estudiante de Doctorado Educación UC

LLECE: Taller de Análisis III

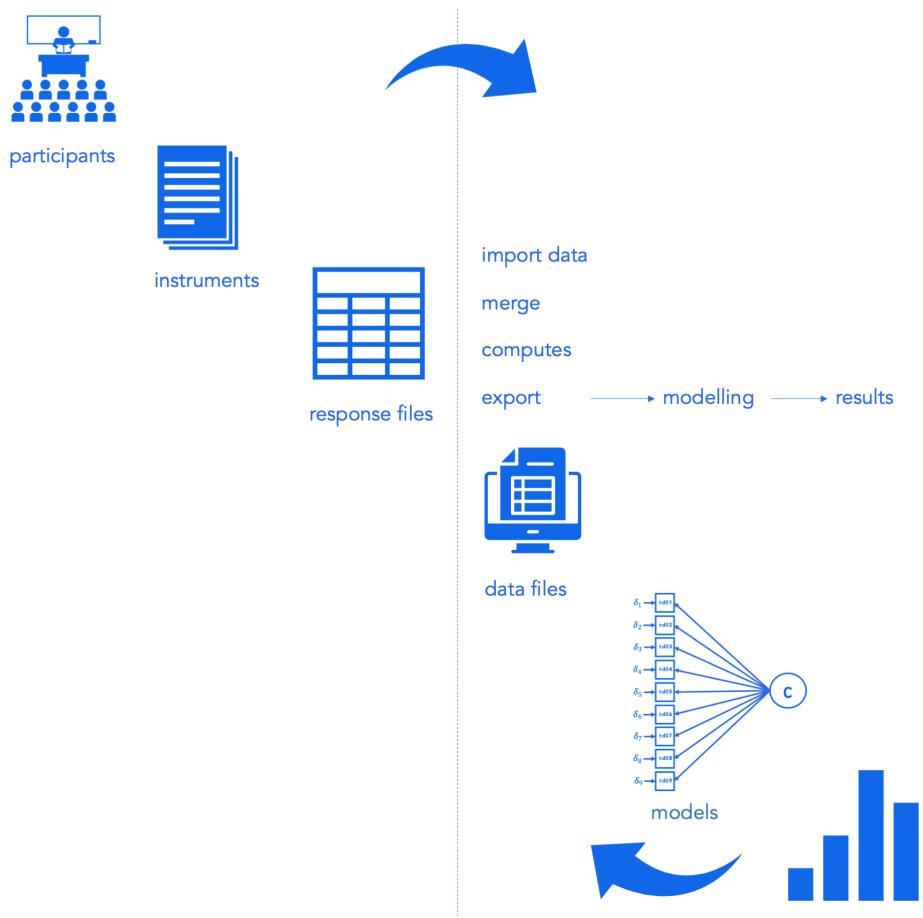
Santiago, Marzo 02 de 2022

Taller

Análisis de datos como desarollo

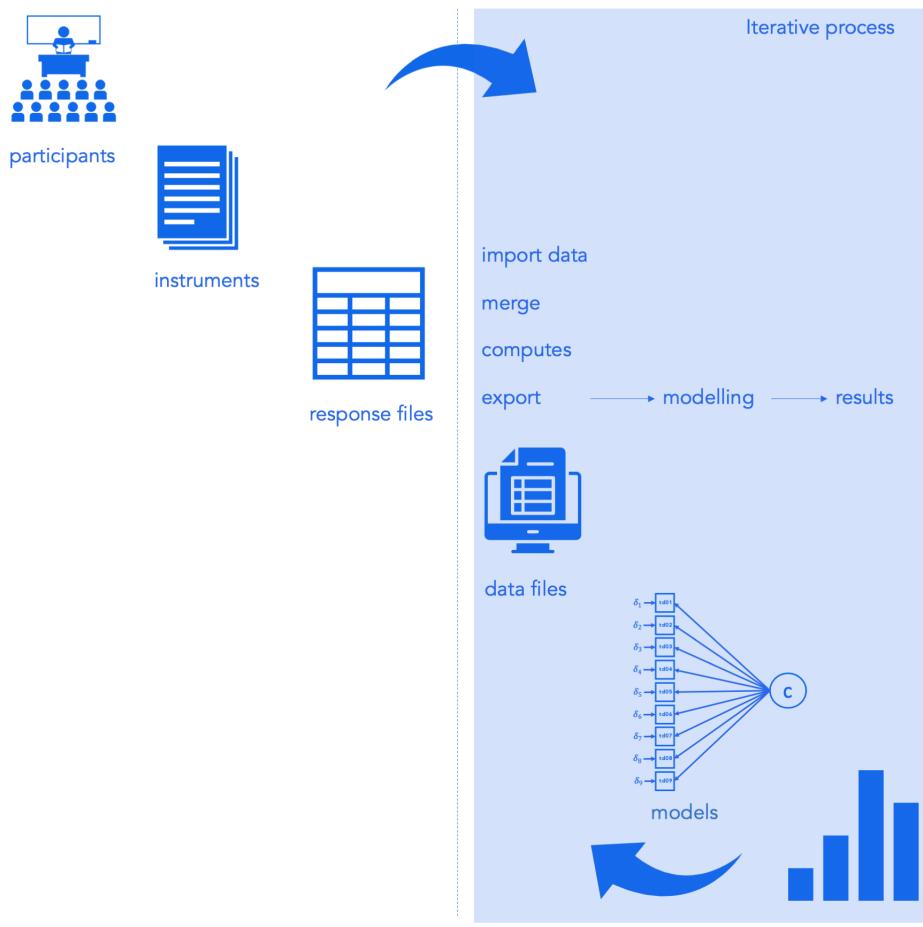
Preparación de datos desde una perspectiva de análisis como desarollo

Análisis de datos como desarrollo

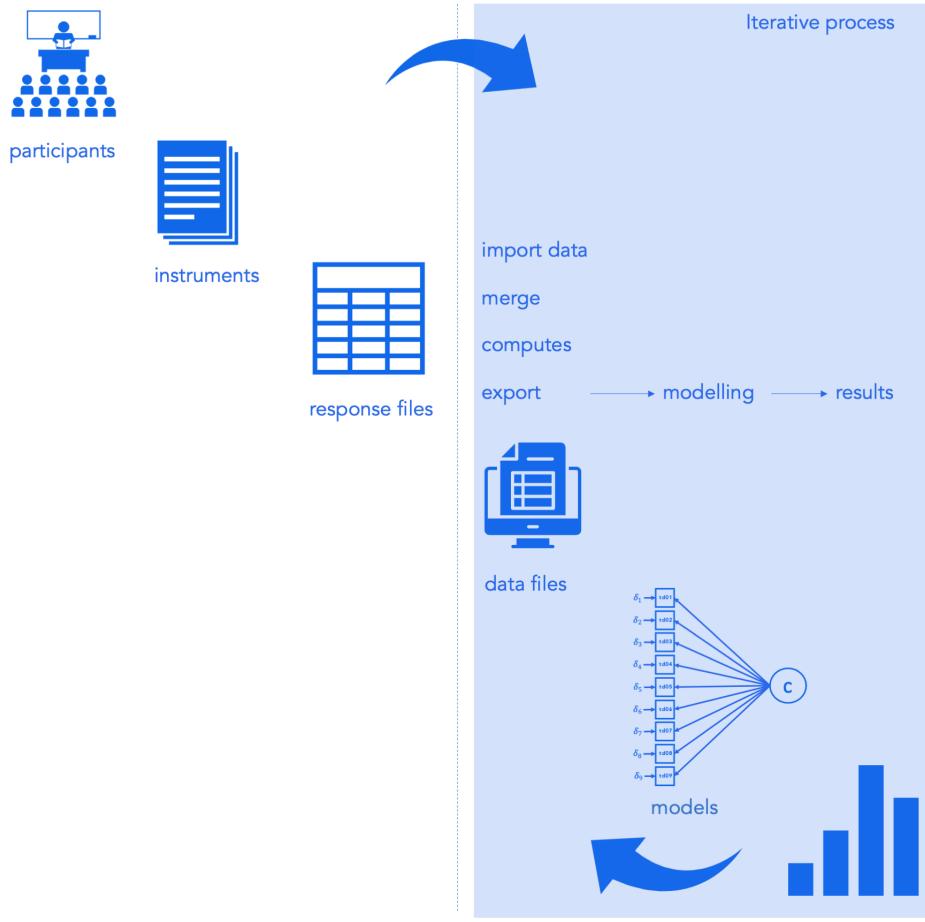


- La generación de resultados, empleando datos de estudios de gran escala, contempla la consideración de **diferentes etapas**, desde la recolección de datos, hasta la presentación de resultados.
- En este proceso se generan una **serie de archivos** por cada instrumento, y por cada país.
- Estas diferentes etapas, pueden ser vistas desde un punto de vista de un **flujo de trabajo**, incluyendo:
 - Levantamiento de datos
 - Digitación
 - Preparación de datos
 - Liberación de datos públicos
 - **Importación de datos**
 - Unión de datos
 - Creación de variables
 - Diseño de datos para modelamiento
 - Exportación de datos
 - **Modelamiento de datos**
 - Tablas y Figuras
 - Presentación de resultados

Análisis de datos como desarrollo



Análisis de datos como desarrollo



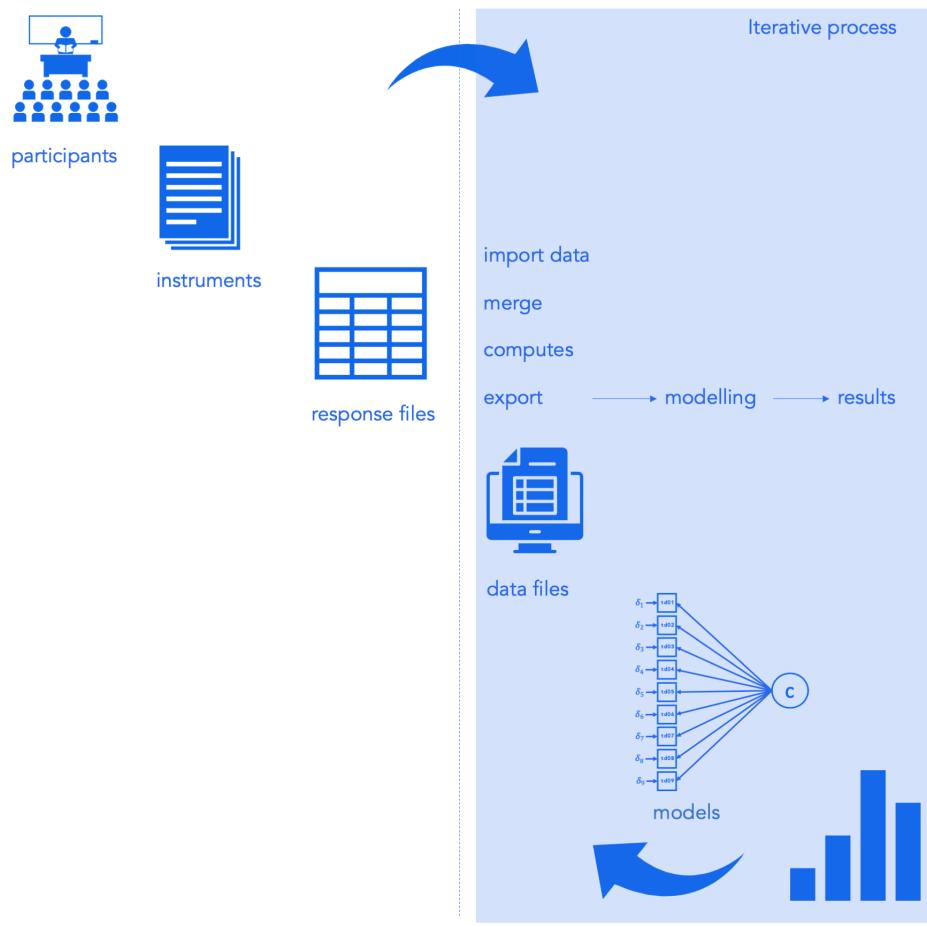
- En este taller, nos vamos a centrar en las etapas de **generación de resultados**. Esto contempla todas las etapas desde la importación de bases de datos, hasta la generación de resultados.
- Podemos pensar el conjunto de etapas entre la importación de datos, hasta la generación de resultados como una **serie de problemas** que requieren ser resueltos por parte de los usuarios de secundarios.
- El presente flujo de trabajo, sigue un principio básico del **opinionated analysis** (Parker, 2017). Este principio consiste en que los análisis de datos se producen como un desarrollo de código, los cuales proponen una forma de trabajo entre otras. Y como tal, no es la forma correcta de construir un proceso de generación de resultados, sino que se presenta como una forma razonable de diseñar un proceso de generación de resultados.
- A continuación, revisaremos las etapas más relevantes de este proceso, cada una como **diferentes problemas a resolver**. Y cada uno de estos problemas, los ilustraremos empleando datos de ERCE 2019.

Taller

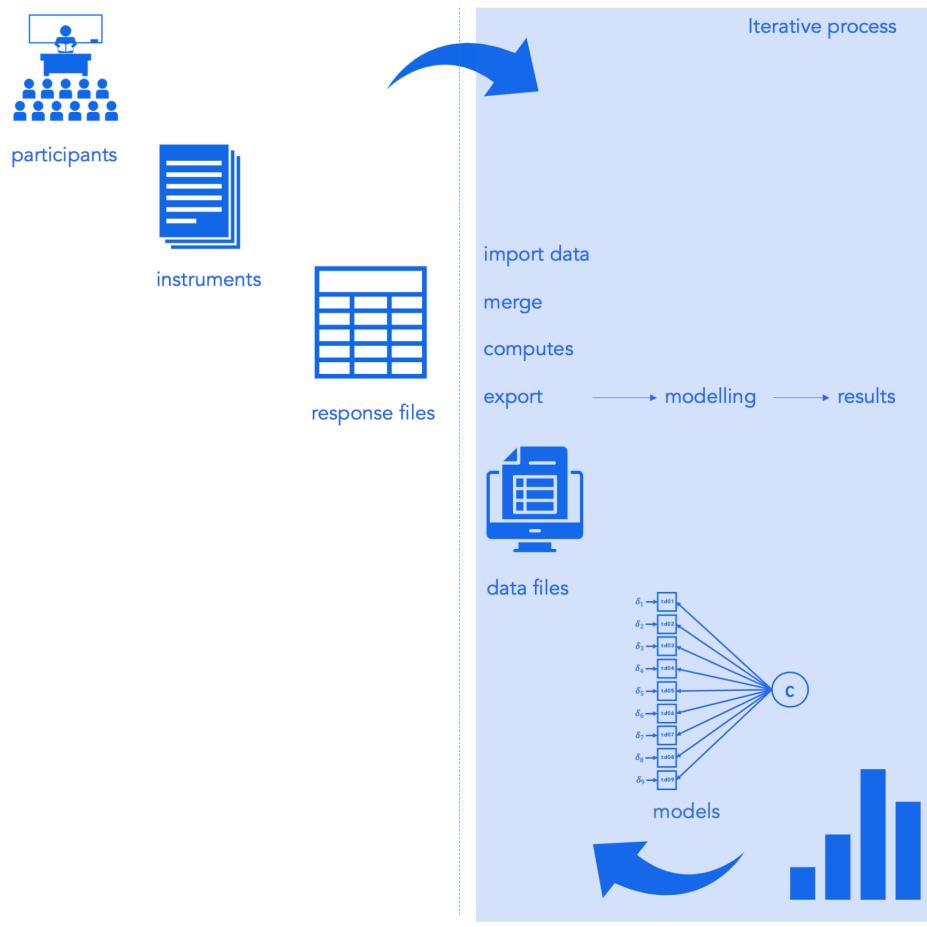
Problemas y soluciones

Generación de resultados como desarollo de una secuencia

Problemas en secuencia: [1] Abrir los datos



Problemas en secuencia: [1] Abrir los datos



- El primer problema a resolver, consiste en como **abrir los datos** que queremos emplear.
 - Algunos estudios de gran escala como ICCS 2016 por ejemplo, proveen de un archivo de datos por país, y por instrumento.
 - En cambio, otros estudios de gran escala proveen de un solo archivo para todos los países, integrando puntajes y respuestas a los cuestionarios de contexto. Este es el caso de PISA 2018.
 - **ERCE 2019** pertenece a este segundo grupo, y los datos se encuentran separados por población de estudiantes (Tercer y Sexto grado), y por instrumento o cuestionario de contexto.

```
# forma rápida de abrir los datos  
erce_a6 <- erce::erce_2019_qa6
```

Listado de archivos

# Base de datos de estudiantes # ERCE_2019_QA3.rds	# Base de datos de familias # ERCE_2019_QF3.rds
# ERCE_2019_QA6.rds	# ERCE_2019_QF6.rds
# Base de datos de profesores # ERCE_2019_QP3.rds	# Base de datos de directores # ERCE_2019_QD3.rds
# ERCE_2019_QP6.rds	# ERCE_2019_QD6.rds

Problemas en secuencia: [2] Contenido de los datos

04. ¿Qué idioma hablan en tu casa la mayor parte del tiempo?

E6IT04

Marca con una X solo una opción.

- 1 <Lengua oficial>
- 2 <Lengua extranjera>
- 3 Otra lengua extranjera.
- 4 <Lengua indígena 1>
- 5 <Lengua indígena 2>
- 6 Otra lengua indígena.

Nota: las opciones de respuesta varían por país. Revisar Anexo 2 de adaptaciones.

```
# muestra de 10 casos de la variable E6IT04
library(dplyr)
erce::erce_2019_qa6 %>%
dplyr::sample_n(10) %>%
dplyr::select(IDSTUD, IDSCHOOL, COUNTRY, E6IT04) %>%
erce::remove_labels() %>%
tibble::as_tibble()
```

```
# output
> library(dplyr)
> erce::erce_2019_qa6 %>%
+ dplyr::sample_n(10) %>%
+ dplyr::select(IDSTUD, IDSCHOOL, COUNTRY, E6IT04) %>%
+ erce::remove_labels() %>%
+ tibble::as_tibble()
# A tibble: 10 × 4
  IDSTUD IDSCHOOL COUNTRY E6IT04
    <dbl>   <dbl> <chr>    <dbl>
1 10300128     1030 COL      1
2 11860225     1186 DOM      1
3 11900107     1190 HND      1
4 11280105     1128 DOM      9
5 11150111     1115 SLV      1
6 10460118     1046 COL      1
7 10940116     1094 PAN      1
8 12070122     1207 GTM      1
9 10620223     1062 URY      1
10 12340107    1234 PAN      1
```

Problemas en secuencia: [2] Contenido de los datos

04. ¿Qué idioma hablan en tu casa la mayor parte del tiempo?

E6IT04

Marca con una X solo una opción.

- 1 <Lengua oficial>
- 2 <Lengua extranjera>
- 3 Otra lengua extranjera.
- 4 <Lengua indígena 1>
- 5 <Lengua indígena 2>
- 6 Otra lengua indígena.

Nota: las opciones de respuesta varían por país. Revisar Anexo 2 de adaptaciones.

```
# muestra de 10 casos de la variable E6IT04
library(dplyr)
erce::erce_2019_qa6 %>%
dplyr::sample_n(10) %>%
dplyr::select(IDSTUD, IDSCHOOL, COUNTRY, E6IT04) %>%
erce::remove_labels() %>%
tibble::as_tibble()
```

```
# output
> library(dplyr)
> erce::erce_2019_qa6 %>%
+ dplyr::sample_n(10) %>%
+ dplyr::select(IDSTUD, IDSCHOOL, COUNTRY, E6IT04) %>%
+ erce::remove_labels() %>%
+ tibble::as_tibble()
# A tibble: 10 × 4
  IDSTUD IDSCHOOL COUNTRY E6IT04
  <dbl>   <dbl> <chr>    <dbl>
1 10300128     1030 COL      1
2 11860225     1186 DOM      1
3 11900107     1190 HND      1
4 11280105     1128 DOM      9
5 11150111     1115 SLV      1
6 10460118     1046 COL      1
7 10940116     1094 PAN      1
8 12070122     1207 GTM      1
9 10620223     1062 URY      1
10 12340107    1234 PAN      1
```

- El segundo problema a resolver, consiste en identificar el **contenido de los datos**.
- Una base de datos, para ser informativa, requiere de **suplementos** para que podamos producir resultados interpretables.
- Es decir, para que podamos contestar a una pregunta del tipo "qué proporción de estudiantes habla en el hogar, el mismo lenguaje del test y los cuestionarios", necesitamos saber **qué pregunta** del instrumento recoge información pertinente a esta pregunta.
- Además requerimos saber como las **respuestas** de las personas fueron **codificadas** en la base de datos.
- Para estos fines, hay dos fuentes de información relevantes. Una es el **manual de usuario**, el cual contiene un libro de códigos plasmado sobre el cuestionario aplicado.
- Y por otro lado, podemos recurrir a la **meta-data** de la base de datos, la cual también nos permite acceder a la respuesta que representa cada valor digitado.

--

```
# mostrar metadata en R
labelled::look_for(erce::erce_2019_qa6)
```

Problemas en secuencia: [3] Variables de diseño

```
# variables de diseño
library(dplyr)
erce:::erce_2019_qa6 %>%
dplyr:::sample_n(10) %>%
dplyr:::select(COUNTRY, STRATA, IDSCHOOL, IDSTUD, WT, WJ, WI) %>%
labelled:::lookfor_to_long_format() %>%
tibble:::as_tibble()

> library(dplyr)
> erce:::erce_2019_qa6 %>%
+ dplyr:::sample_n(10) %>%
+ dplyr:::select(COUNTRY, STRATA, IDSCHOOL, IDSTUD, WT, WJ, WI) %>%
+ labelled:::lookfor_to_long_format() %>%
+ tibble:::as_tibble()
# A tibble: 10 × 7
  COUNTRY    STRATA IDSCHOOL   IDSTUD     WT     WJ     WI
  <chr>      <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
1 PER [Peru] 66042121 1214 12140101 132.  123.   1.08 
2 NIC [Nicaragua] 65582111 1217 12170238  4.65  1.33  3.49 
3 ECU [Ecuador] 62181132 1111 11110110 24.7  24.7   1    
4 BRA [Brasil] 60762111 1131 11310226 634.  245.   2.59 
5 PER [Peru] 66042111 1180 11800321 60.5  12.1    5    
6 MEX [Mexico] 64841111 1032 10320125 395.  124.   3.18 
7 NIC [Nicaragua] 65582112 1245 12450102 33.0  30.9  1.07 
8 ECU [Ecuador] 62182111 1178 11780128 52.5  52.5   1    
9 NIC [Nicaragua] 65582111 1218 12180323  5.40  1.51  3.58 
10 ARG [Argentina] 60322111 1062 10620205 237.  112.   2.11
```

```
# muestra de valores de variables de diseño
library(dplyr)
erce:::erce_2019_qa6 %>%
dplyr:::sample_n(10) %>%
dplyr:::select(COUNTRY, STRATA, IDSCHOOL, IDSTUD, WT, WJ, WI) %>%
erce:::remove_labels() %>%
tibble:::as_tibble()
```

```
> library(dplyr)
> erce:::erce_2019_qa6 %>%
+ dplyr:::sample_n(10) %>%
+ dplyr:::select(COUNTRY, STRATA, IDSCHOOL, IDSTUD, WT, WJ, WI) %>%
+ erce:::remove_labels() %>%
+ tibble:::as_tibble()
# A tibble: 10 × 7
  COUNTRY    STRATA IDSCHOOL   IDSTUD     WT     WJ     WI
  <chr>      <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
1 PRY 66002131 1175 11750117 17.4  5.81  3.00 
2 PRY 66002131 1062 10620222 12.8  4.28  3.00 
3 COL 61702111 1024 10240205 174.  81.5  2.13 
4 ECU 62182111 1202 12020138 32.1  15.6  2.05 
5 HND 63402111 1041 10410126 17.4  14.6  1.19 
6 NIC 65582112 1294 12940114  4.52  3.32  1.36 
7 PER 66042121 1218 12180106 197.  167.  1.18 
8 PAN 65912112 1225 12250105 17.4  17.4   1    
9 NIC 65582121 1037 10370221 10.2  4.90  2.09 
10 BRA 60762111 1140 11400103 371.  152.  2.44
```

Problemas en secuencia: [3] Variables de diseño

```
# variables de diseño
library(dplyr)
erce:::erce_2019_qa6 %>%
dplyr:::sample_n(10) %>%
dplyr:::select(COUNTRY, STRATA, IDSCHOOL, IDSTUD, WT, WJ, WI) %>%
labelled:::lookfor_to_long_format() %>%
tibble:::as_tibble()
```

```
> library(dplyr)
> erce:::erce_2019_qa6 %>%
+ dplyr:::sample_n(10) %>%
+ dplyr:::select(COUNTRY, STRATA, IDSCHOOL, IDSTUD, WT, WJ, WI) %>%
+ labelled:::lookfor_to_long_format() %>%
+ tibble:::as_tibble()
# A tibble: 10 × 7
  COUNTRY    STRATA IDSCHOOL IDSTUD     WT     WJ     WI
  <chr>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
1 PER [Peru]  66042121  1214 12140101 132.  123.  1.08 
2 NIC [Nicaragua] 65582111  1217 12170238  4.65  1.33  3.49 
3 ECU [Ecuador]  62181132  1111 11110110 24.7  24.7  1      
4 BRA [Brasil]  60762111  1131 11310226 634.  245.  2.59 
5 PER [Peru]   66042111  1180 11800321 60.5  12.1  5      
6 MEX [Mexico] 64841111  1032 10320125 395.  124.  3.18 
7 NIC [Nicaragua] 65582112  1245 12450102 33.0  30.9  1.07 
8 ECU [Ecuador]  62182111  1178 11780128 52.5  52.5  1      
9 NIC [Nicaragua] 65582111  1218 12180323  5.40  1.51  3.58 
10 ARG [Argentina] 60322111 1062 10620205 237.  112.  2.11
```

```
# muestra de valores de variables de diseño
library(dplyr)
erce:::erce_2019_qa6 %>%
dplyr:::sample_n(10) %>%
dplyr:::select(COUNTRY, STRATA, IDSCHOOL, IDSTUD, WT, WJ, WI) %>%
erce:::remove_labels() %>%
tibble:::as_tibble()
```

```
> library(dplyr)
> erce:::erce_2019_qa6 %>%
+ dplyr:::sample_n(10) %>%
+ dplyr:::select(COUNTRY, STRATA, IDSCHOOL, IDSTUD, WT, WJ, WI) %>%
+ erce:::remove_labels() %>%
+ tibble:::as_tibble()
# A tibble: 10 × 7
  COUNTRY    STRATA IDSCHOOL IDSTUD     WT     WJ     WI
  <chr>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
1 PRY  66002131  1175 11750117 17.4  5.81  3.00 
2 PRY  66002131  1062 10620222 12.8  4.28  3.00 
3 COL  61702111  1024 10240205 174.  81.5  2.13 
4 ECU  62182111  1202 12020138 32.1  15.6  2.05 
5 HND  63402111  1041 10410126 17.4  14.6  1.19 
6 NIC  65582112  1294 12940114  4.52  3.32  1.36 
7 PER  66042121  1218 12180106 197.  167.  1.18 
8 PAN  65912112  1225 12250105 17.4  17.4  1      
9 NIC  65582121  1037 10370221 10.2  4.90  2.09 
10 BRA 60762111 1140 11400103 371.  152.  2.44
```

- Dependiendo de qué tipo de pregunta o preguntas queramos abordar, necesitamos saber como manejar las **variables de diseño del estudio**.
 - El escenario más sencillo, es cuando queremos trabajar con los datos de un sólo país.
 - Sin embargo, cuando queremos producir resultados empleando más de un país, es posible que sea necesario tratar a las variables de escuelas, estratos, y pesos de diferente forma, para producir resultados para la región.
 - De la misma forma, si queremos producir resultados combinando a la población de estudiantes de Tercer y Sexto grado, debemos tomar decisiones respecto a cómo tratar a las variables de diseño (e.g., escuelas, estratos, y pesos muestrales).
 - Finalmente, si queremos vincular resultados de TERCE y ERCE, tambien debemos tomar decisiones de quéharemos con las variables de diseño (e.g., escuelas, estratos, y pesos muestrales).

Nota: Los estudios de gran escala permiten realizar comparaciones con datos que emplean un diseño similar, permitiendo comparaciones con otros países, con otros grados, y con datos previos. Pero se requiere realizar algún tratamiento sobre las variables de diseño, de modo que el cálculo de errores, y la expansión de observaciones sea adecuada.

Problemas en secuencia: [4] Dirección de los valores de respuesta

33. ¿Con cuáles de estos servicios cuenta la escuela?

Frente a cada uno de los servicios señalados, marque con una X solo una opción de respuesta.

		Si ₁	No ₂
DDIT33_01	Luz eléctrica.	<input type="radio"/>	<input type="radio"/>
DDIT33_02	Agua potable.	<input type="radio"/>	<input type="radio"/>
DDIT33_03	Desagüe o alcantarillado.	<input type="radio"/>	<input type="radio"/>
DDIT33_04	Teléfono.	<input type="radio"/>	<input type="radio"/>
DDIT33_05	Laboratorio móvil de computación.	<input type="radio"/>	<input type="radio"/>
DDIT33_06	Baños en buen estado.	<input type="radio"/>	<input type="radio"/>
DDIT33_07	Conexión a Internet.	<input type="radio"/>	<input type="radio"/>
DDIT33_08	Recolección de basura.	<input type="radio"/>	<input type="radio"/>
DDIT33_09	Transporte de estudiantes (gratuito para las familias).	<input type="radio"/>	<input type="radio"/>

```
# muestra de valores de variables de diseño
library(dplyr)
erce::erce_2019_qd6 %>%
dplyr::select(DDIT33_07) %>%
labelled::lookfor()
```

```
pos variable label col_type values
1 DDIT33_07 Conexión a Internet. dbl+lbl [1] Sí
                                         [2] No
```

Problemas en secuencia: [4] Dirección de los valores de respuesta

33. ¿Con cuáles de estos servicios cuenta la escuela?

Frente a cada uno de los servicios señalados, marque con una X solo una opción de respuesta.

		Si ₁	No ₂
DDIT33_01	Luz eléctrica.	<input type="radio"/>	<input type="radio"/>
DDIT33_02	Agua potable.	<input type="radio"/>	<input type="radio"/>
DDIT33_03	Desagüe o alcantarillado.	<input type="radio"/>	<input type="radio"/>
DDIT33_04	Teléfono.	<input type="radio"/>	<input type="radio"/>
DDIT33_05	Laboratorio móvil de computación.	<input type="radio"/>	<input type="radio"/>
DDIT33_06	Baños en buen estado.	<input type="radio"/>	<input type="radio"/>
DDIT33_07	Conexión a Internet.	<input type="radio"/>	<input type="radio"/>
DDIT33_08	Recolección de basura.	<input type="radio"/>	<input type="radio"/>
DDIT33_09	Transporte de estudiantes (gratuito para las familias).	<input type="radio"/>	<input type="radio"/>

```
# muestra de valores de variables de diseño
library(dplyr)
erce:::erce_2019_qd6 %>%
dplyr:::select(DDIT33_07) %>%
labelled:::lookfor()
```

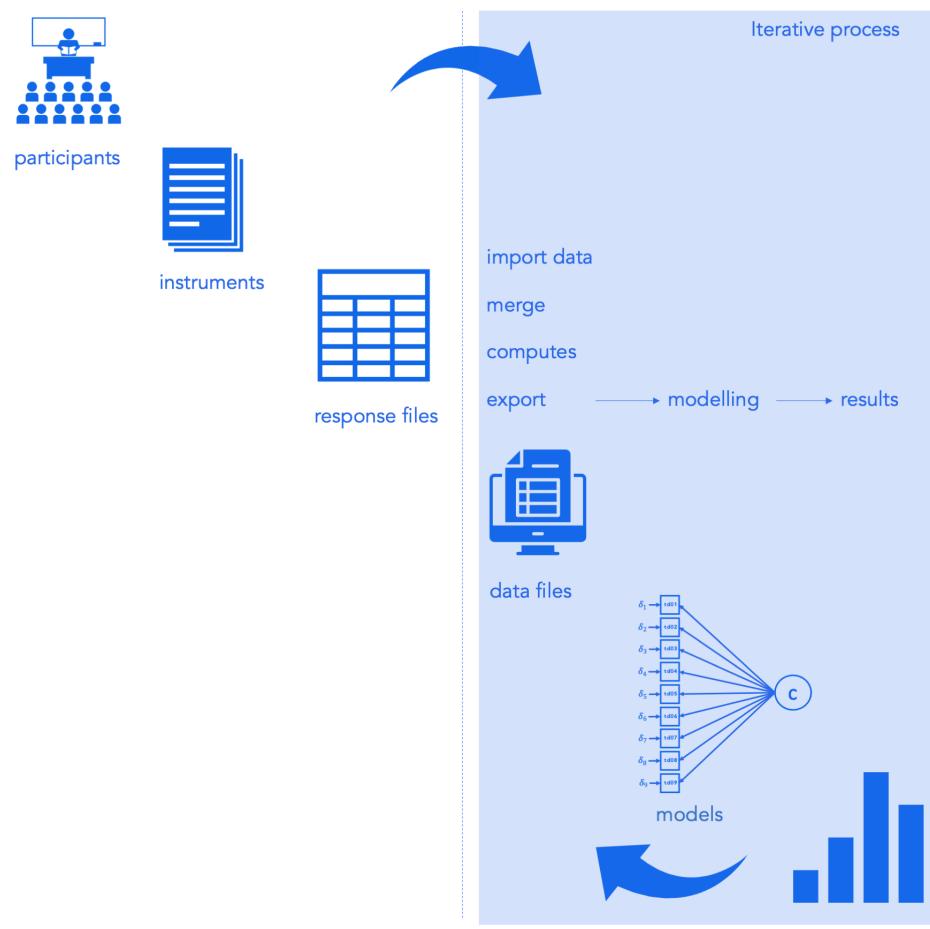
```
pos variable label col_type values
1 DDIT33_07 Conexión a Internet. dbl+lbl [1] Sí
[2] No
```

- Por convención, **los valores de digitación de respuestas observadas se asignan en secuencias de izquierda a derecha**. De este modo, la primera opción de respuesta se digita como 1, la segunda como 2, y así sucesivamente.

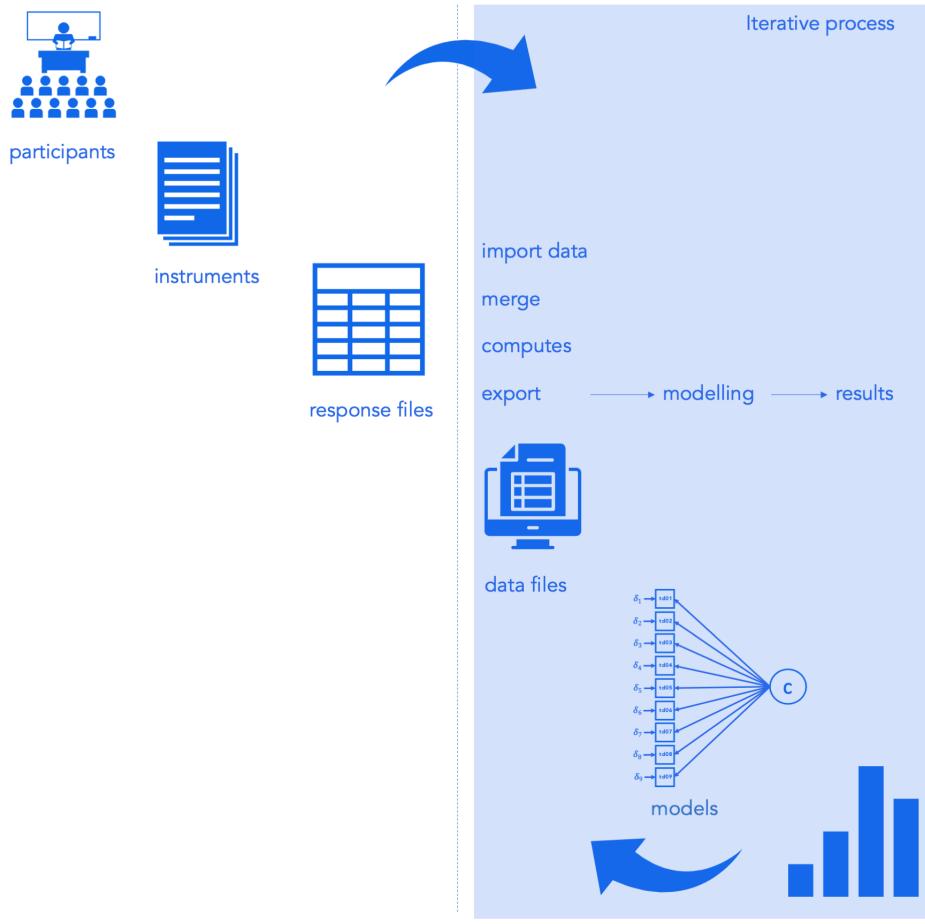
- Considerando la regla anterior, es posible que, para extraer la información deseada respecto a una pregunta, el usuario secundario requiera **crear variables nuevas** empleando la información original.
- La variable **DDIT33_07**, nos permite obtener que porcentaje de estudiantes, en un país posee conexión a internet en sus escuelas.
- Si bien la variable **DDIT33_07** contiene la información de interés, la manera en que están digitados los valores originales, no necesariamente nos facilita el trabajo para obtener el porcentaje de interés.
- De este modo, es común que previo a la generación de resultados, los usuarios secundarios requieran **transformar las respuestas digitadas**, a variables que entreguen la información buscada por las preguntas de interés.

Nota: En este ejemplo, una forma de generar resultados consistiría en convertir los valores 2, en valores 0, generando otra variable, y con esta nueva variable calcular los porcentajes esperados.

Problemas en secuencia: [5] Transferibilidad



Problemas en secuencia: [5] Transferibilidad



- Es común que la realización de ciertos análisis estadísticos requieran ser realizados en **diferentes software**.
- También es común que los usuarios secundarios **colaboren con otras personas**, y estas otras personas empleen software diferentes a las que se encuentra empleando la primera persona.
- Le llamaremos problema de **transferibilidad**, al escenario en que esté en juego qué tan transferibles son los datos originales a otras plataformas de análisis de datos.
- Este problema, se resuelve comúnmente, mediante la **exportación de datos** de un software a otro, empleando formatos determinados.
- De este modo, es común que previo a la generación de resultados, los usuarios secundarios requieran **transformar las respuestas digitadas**, a variables que entreguen la información buscada por las preguntas de interés.
- La importación de datos, la identificación del contenido de los datos, la identificación de las variables de diseño, la preparación de variables, y la exportación de datos, previos a la generación de resultados, son todas etapas que las podemos considerar parte de un **proceso iterativo**. En otras palabras, que lo haremos no solo una vez, sino varias veces.

A continuación, revisaremos algunos ejemplos de estos procesos, empleando datos de ERCE 2019.

Problemas y soluciones

Abrir datos

Problema 1: Cómo cargar los datos de ERCE 2019 en R

Problema 1: Abrir los datos

La manera más sencilla de abrir los datos en R, es emplear la librería.

```
#-----  
# Abrir los datos desde la librería erce  
#-----  
  
erce_a6 <- erce::erce_2019_qa6
```

Otra forma consiste en abrir los datos empleando los archivos .rds

```
#-----  
# abrir los datos empleando el archivo rds  
#-----  
  
data_folder <- '/Users/d/ERCE 2019/'  
  
erce_a6 <- readRDS(paste0(data_folder, 'ERCE_2019_QA6.rds'))
```

En R también es posible abrir los datos en formato STATA .dta

```
#-----  
# abrir los datos de formato STATA  
#-----  
  
data_folder <- '/Users/d/ERCE 2019/'  
  
erce_a6 <- haven::read_dta(paste0(data_folder, 'ERCE_2019_QA6.dta'))
```

Se procede de forma similar para abrir los datos en formato SPSS .sav

```
#-----  
# abrir los datos de formato SPSS  
#-----  
  
data_folder <- '/Users/d/ERCE 2019/'  
  
erce_a6 <- haven::read_sav(paste0(data_folder, 'ERCE_2019_QA6.sav'))
```

Problema 1: Abrir los datos

La manera más sencilla de abrir los datos en R, es emplear la librería.

```
#-----  
# Abrir los datos desde la librería erce  
#-----  
erce_a6 <- erce::erce_2019_qa6
```

Otra forma consiste en abrir los datos empleando los archivos **.rds**

```
#-----  
# abrir los datos empleando el archivo rds  
#-----  
  
data_folder <- '/Users/d/ERCE 2019/'  
erce_a6 <- readRDS(paste0(data_folder, 'ERCE_2019_QA6.rds'))
```

En R también es posible abrir los datos en formato STATA **.dta**

```
#-----  
# abrir los datos de formato STATA  
#-----  
  
data_folder <- '/Users/d/ERCE 2019/'  
erce_a6 <- haven::read_dta(paste0(data_folder, 'ERCE_2019_QA6.dta'))
```

Se procede de forma similar para abrir los datos en formato SPSS **.sav**

```
#-----  
# abrir los datos de formato SPSS  
#-----  
  
data_folder <- '/Users/d/ERCE 2019/'  
erce_a6 <- haven::read_sav(paste0(data_folder, 'ERCE_2019_QA6.sav'))
```

La primera forma de abrir los datos es desde la librería **erce** que utilizaremos a lo largo de este taller. La ventaja que posee esta forma, es que no necesitamos ubicar donde está la base de datos, porque se encuentra accesible por el entorno R, a través de las funciones de la librería.

La segunda forma, requiere que definamos el objeto **data_folder**. Esta es una dirección en sus computadores, donde los archivos de datos se encuentran alojados.

La tercera y cuarta forma son equivalentes a la segunda forma. La única diferencia es que se encuentran abriendo los datos en otros formatos, en formatos STATA y SPSS respectivamente.

Problema 1: Abrir los datos

La manera más sencilla de abrir los datos en R, es emplear la librería.

```
#-----  
# Abrir los datos desde la librería erce  
#-----  
  
erce_a6 <- erce::erce_2019_qa6
```

Otra forma consiste en abrir los datos empleando los archivos .rds

```
#-----  
# abrir los datos empleando el archivo rds  
#-----  
  
data_folder <- '/Users/d/ERCE 2019/'  
  
erce_a6 <- readRDS(paste0(data_folder, 'ERCE_2019_QA6.rds'))
```

En R también es posible abrir los datos en formato STATA .dta

```
#-----  
# abrir los datos de formato STATA  
#-----  
  
data_folder <- '/Users/d/ERCE 2019/'  
  
erce_a6 <- haven::read_dta(paste0(data_folder, 'ERCE_2019_QA6.dta'))
```

Se procede de forma similar para abrir los datos en formato SPSS .sav

```
#-----  
# abrir los datos de formato SPSS  
#-----  
  
data_folder <- '/Users/d/ERCE 2019/'  
  
erce_a6 <- haven::read_sav(paste0(data_folder, 'ERCE_2019_QA6.sav'))
```

Problema 1: Abrir los datos

La manera más sencilla de abrir los datos en R, es emplear la librería.

```
#--  
# Abrir los datos desde la librería erce  
#--  
erce_a6 <- erce::erce_2019_qa6
```

Otra forma consiste en abrir los datos empleando los archivos .rds

```
#--  
# abrir los datos empleando el archivo rds  
#--  
  
data_folder <- '/Users/d/ERCE 2019/'  
  
erce_a6 <- readRDS(paste0(data_folder, 'ERCE_2019_QA6.rds'))
```

En R también es posible abrir los datos en formato STATA .dta

```
#--  
# abrir los datos de formato STATA  
#--  
  
data_folder <- '/Users/d/ERCE 2019/'  
  
erce_a6 <- haven::read_dta(paste0(data_folder, 'ERCE_2019_QA6.dta'))
```

Se procede de forma similar para abrir los datos en formato SPSS .sav

```
#--  
# abrir los datos de formato SPSS  
#--  
  
data_folder <- '/Users/d/ERCE 2019/'  
  
erce_a6 <- haven::read_sav(paste0(data_folder, 'ERCE_2019_QA6.sav'))
```

Una vez que los datos se encuentran cargados en el entorno R, podemos revisar su contenido.

```
#--  
# abrir datos  
#--  
  
erce_a6 <- erce::erce_2019_qa6  
  
#--  
# revisar su contenido  
#--  
  
library(dplyr)  
erce_a6 %>%  
dplyr::select(IDSTUD:E6IT10) %>%  
dplyr::glimpse()  
  
# Nota: seleccionamos algunas variables,  
# con la linea 'dplyr::select(IDSTUD:E6IT10) %>%'  
# solo para que el output sea visible en pantalla.
```

```
> library(dplyr)  
> erce_a6 %>%  
+ dplyr::select(IDSTUD:E6IT10) %>% #<<  
+ dplyr::glimpse()  
Rows: 80,827  
Columns: 29  
$ IDSTUD <dbl> 10010502, 10010503, 10010504, 10010505, 10010506, 10010507, ...  
$ IDCLASS <dbl> 100105, 100105, 100105, 100105, 100105, NA, 100105, 100105, ...  
$ IDSCHOOL <dbl> 1001, 1001, 1001, 1001, 1001, 1001, 1001, 1001, 1001, 1001, ...  
$ IDCNTRY <dbl+lbl> 170, 170, 170, 170, 170, 170, 170, 170, 170, 170, 170, 1...  
$ COUNTRY <chr+lbl> "COL", "COL", "COL", "COL", "COL", "COL", "COL", "COL", "COL", ...  
$ STRATA <dbl> 61702111, 61702111, 61702111, 61702111, 61702111, 61702111, ...  
$ GRADE <dbl+lbl> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,...  
$ QA6 <dbl+lbl> 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...  
$ WT <dbl> 210.6399906, 210.6399906, 210.6399906, 210.6399906, 210.6399906,...  
$ WS <dbl> 0.2484165221, 0.2484165221, 0.2484165221, 0.2484165221, 0.24...  
$ WI <dbl> 8.272727275, 8.272727275, 8.272727275, 8.272727275, 8.272727...  
$ WJ <dbl> 25.46197688, 25.46197688, 25.46197688, 25.46197688, 25.46197...  
$ E6IT01 <dbl+lbl> 4, 5, 3, 5, 3, NA, 2, 6, 5, 1, 3, 2, 2, 5, ...  
$ E6IT02 <dbl+lbl> 2, 2, 2, 2, NA, 1, 2, 2, 1, 2, 1, 1, 2, ...  
$ E6IT03_01 <dbl+lbl> NA, 1, 2, 2, 1, NA, 2, NA, 2, 1, 2, 1, 1, 1, ...  
$ E6IT03_02 <dbl+lbl> 1, 2, 2, 1, 2, NA, 1, 2, 1, 1, 2, 1, 2, 2, ...  
$ E6IT03_03 <dbl+lbl> NA, 2, 2, 2, 1, NA, 1, NA, 1, 2, 2, 1, 1, 1, ...  
$ E6IT03_04 <dbl+lbl> NA, 2, 1, 2, 1, NA, 2, 2, 1, 2, 1, 1, 1, 2, ...  
$ E6IT03_05 <dbl+lbl> NA, 2, 2, 2, 1, NA, 2, 2, 1, 2, 1, 2, 2, NA, 1, 1, 2, ...  
$ E6IT03_06 <dbl+lbl> NA, 1, 2, 2, 1, NA, 1, 2, 1, 2, 1, NA, 1, 2, ...  
$ E6IT04 <dbl+lbl> 1, 1, 1, 1, 1, NA, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...  
$ E6IT05 <dbl+lbl> 2, 2, 2, 2, 2, NA, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, ...  
$ E6IT05A <dbl+lbl> NA, NA, NA, NA, 6, NA, NA, NA, NA, NA, NA, NA, NA, ...  
$ E6IT06 <dbl+lbl> 1, 1, 1, 1, 1, NA, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...  
$ E6IT06A <dbl> 99, 99, 99, 99, 99, NA, 99, 15, 99, 99, 99, 99, 99, 99, ...  
$ E6IT07 <dbl+lbl> NA, 1, 1, 1, 1, NA, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...  
$ E6IT08 <dbl+lbl> NA, 1, 1, 1, 1, NA, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...  
$ E6IT09 <dbl+lbl> 99, 5, 2, 7, 3, NA, 3, 6, 2, 7, 3, 3, 7, 2, 2, ...  
$ E6IT10 <dbl+lbl> 2, 2, 1, 2, 2, NA, 2, 4, 3, 2, 3, 4, 3, 2, ...
```

Problemas y soluciones

Inspeccionar el contenido de los datos

Problema 2: Cómo saber cuál es el contenido de los datos

Problema 2: Inspeccionar el contenido de los datos

- Una vez que los datos se encuentran abiertos o cargados, lo siguiente que necesitamos es poder **ver su contenido**.
- Existen **diferentes formas** de inspeccionar los datos en R, una vez que estos se encuentran cargados en la sesión.

Problema 2: Inspeccionar el contenido de los datos

- Una vez que los datos se encuentran abiertos o cargados, lo siguiente que necesitamos es poder **ver su contenido**.
- Existen **diferentes formas** de inspeccionar los datos en R, una vez que estos se encuentran cargados en la sesión.
- Una forma global de inspeccionar los datos, es emplear el comando **dplyr::glimpse**. Este nos entrega el listado de variables, su tipo, y una muestra de los valores contenidos en cada columna de la base de datos.

Problema 2: Inspeccionar el contenido de los datos

- Una vez que los datos se encuentran abiertos o cargados, lo siguiente que necesitamos es poder **ver su contenido**.
- Existen **diferentes formas** de inspeccionar los datos en R, una vez que estos se encuentran cargados en la sesión.
- Una forma global de inspeccionar los datos, es emplear el comando **dplyr::glimpse**. Este nos entrega el listado de variables, su tipo, y una muestra de los valores contenidos en cada columna de la base de datos.

```
#--  
# abrir datos  
#--  
  
erce_a6 <- erce::erce_2019_qa6  
  
#--  
# revisar contenido con dplyr::glimpse()  
#--  
  
library(dplyr)  
erce_a6 %>%  
dplyr::glimpse()  
  
  
> library(dplyr)  
> erce_a6 %>%  
+ dplyr::glimpse()  
Rows: 80,827  
Columns: 303  
$ IDSTUD <dbl> 10010502, 10010503, 10010504, 10010505, 10010506, 10010507, ...  
$ IDCLASS <dbl> 100105, 100105, 100105, 100105, NA, 100105, 100105, ...  
$ IDSCHOOL <dbl> 1001, 1001, 1001, 1001, 1001, 1001, 1001, 1001, 1001, ...  
$ IDCNTRY <dbl+lbl> 170, 170, 170, 170, 170, 170, 170, 170, 170, 170, 170, 170, 1...  
$ COUNTRY <chr+lbl> "COL", "COL", "COL", "COL", "COL", "COL", "COL", "COL", ...  
$ STRATA <dbl> 61702111, 61702111, 61702111, 61702111, 61702111, 61702111, ...  
$ GRADE <dbl+lbl> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, ...  
$ QA6 <dbl+lbl> 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...  
$ WT <dbl> 210.6399906, 210.6399906, 210.6399906, 210.6399906, 210.6399...  
$ WS <dbl> 0.2484165221, 0.2484165221, 0.2484165221, 0.2484165221, 0.24...  
$ WI <dbl> 8.272727275, 8.272727275, 8.272727275, 8.272727275, 8.272727...  
$ WJ <dbl> 25.46197688, 25.46197688, 25.46197688, 25.46197688, 25.46197...  
$ E6IT01 <dbl+lbl> 4, 5, 3, 5, 3, NA, 2, 6, 5, 1, 3, 2, 2, 5, ...  
$ E6IT02 <dbl+lbl> 2, 2, 2, 2, 2, NA, 1, 2, 2, 1, 2, 1, 1, 2, ...  
$ E6IT03_01 <dbl+lbl> NA, 1, 2, 2, 1, NA, 2, NA, 2, 1, 2, 1, 1, 1, ...  
$ E6IT03_02 <dbl+lbl> 1, 2, 2, 1, 2, NA, 1, 2, 1, 1, 2, 1, 2, 2, ...  
$ E6IT03_03 <dbl+lbl> NA, 2, 2, 2, 1, NA, 1, NA, 1, 2, 2, 1, 1, 1, ...  
$ E6IT03_04 <dbl+lbl> NA, 2, 1, 2, 1, NA, 2, 2, 1, 2, 1, 1, 1, 2, ...  
$ E6IT03_05 <dbl+lbl> NA, 2, 2, 2, 1, NA, 2, 2, 1, 2, 2, NA, 1, 2, ...  
$ E6IT03_06 <dbl+lbl> NA, 1, 2, 2, 1, NA, 1, 2, 1, 2, 1, NA, 1, 2, ...  
$ E6IT04 <dbl+lbl> 1, 1, 1, 1, NA, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...  
[...]
```

Problema 2: Inspeccionar el contenido de los datos

- Una vez que los datos se encuentran abiertos o cargados, lo siguiente que necesitamos es poder **ver su contenido**.
- Existen **diferentes formas** de inspeccionar los datos en R, una vez que estos se encuentran cargados en la sesión.
- Una forma global de inspeccionar los datos, es emplear el comando **dplyr::glimpse**. Este nos entrega el listado de variables, su tipo, y una muestra de los valores contenidos en cada columna de la base de datos.
- Otra forma de inspeccionar los datos, es emplear las funciones de la librería **labelled**. Por ejemplo, empleando el comando **labelled::lookfor** podemos acceder a la meta data de los datos. Una vez aplicado este comando, podemos acceder a la descripción de las variables. Y, para el caso de las variables categóricas, al significado de los valores contenidos en cada columna (e.g., IDCNTRY, COUNTRY).

Problema 2: Inspeccionar el contenido de los datos

- Una vez que los datos se encuentran abiertos o cargados, lo siguiente que necesitamos es poder **ver su contenido**.
 - Existen **diferentes formas** de inspeccionar los datos en R, una vez que estos se encuentran cargados en la sesión.
 - Una forma global de inspeccionar los datos, es emplear el comando **dplyr::glimpse**. Este nos entrega el listado de variables, su tipo, y una muestra de los valores contenidos en cada columna de la base de datos.
 - Otra forma de inspeccionar los datos, es emplear las funciones de la librería **labelled**. Por ejemplo, empleando el comando **labelled::lookfor** podemos acceder a la meta data de los datos. Una vez aplicado este comando, podemos acceder a la descripción de las variables. Y, para el caso de las variables categóricas, al significado de los valores contenidos en cada columna (e.g., IDCNTRY, COUNTRY).

[...]

Problema 2: Inspeccionar el contenido de los datos

- Una vez que los datos se encuentran abiertos o cargados, lo siguiente que necesitamos es poder **ver su contenido**.
- Existen **diferentes formas** de inspeccionar los datos en R, una vez que estos se encuentran cargados en la sesión.
- Una forma global de inspeccionar los datos, es emplear el comando **dplyr::glimpse**. Este nos entrega el listado de variables, su tipo, y una muestra de los valores contenidos en cada columna de la base de datos.
- Otra forma de inspeccionar los datos, es emplear las funciones de la librería **labelled**. Por ejemplo, empleando el comando **labelled::lookfor** podemos acceder a la meta data de los datos. Una vez aplicado este comando, podemos acceder a la descripción de las variables. Y, para el caso de las variables categóricas, al significado de los valores contenidos en cada columna (e.g., IDCNTRY, COUNTRY).
- También es posible generar un libro de códigos con **labelled**, y guardar este libro de códigos en un archivo excel.

Problema 2: Inspeccionar el contenido de los datos

- Una vez que los datos se encuentran abiertos o cargados, lo siguiente que necesitamos es poder **ver su contenido**.
- Existen **diferentes formas** de inspeccionar los datos en R, una vez que estos se encuentran cargados en la sesión.
- Una forma global de inspeccionar los datos, es emplear el comando **dplyr::glimpse**. Este nos entrega el listado de variables, su tipo, y una muestra de los valores contenidos en cada columna de la base de datos.
- Otra forma de inspeccionar los datos, es emplear las funciones de la librería **labelled**. Por ejemplo, empleando el comando **labelled::lookfor** podemos acceder a la meta data de los datos. Una vez aplicado este comando, podemos acceder a la descripción de las variables. Y, para el caso de las variables categóricas, al significado de los valores contenidos en cada columna (e.g., IDCTRY, COUNTRY).
- También es posible generar un libro de códigos con **labelled**, y guardar este libro de códigos en un archivo excel.

```
#--  
# abrir datos  
#--  
  
erce_a6 <- erce::erce_2019_qa6  
  
#--  
# guardar libro de códigos  
#--  
  
library(dplyr)  
erce_a6 %>%  
  labelled::lookfor() %>%  
  labelled::lookfor_to_long_format() %>%  
  tibble::as_tibble() %>%  
  openxlsx::write.xlsx('erce_a6_codebook.xlsx', overwrite = TRUE)
```

Problema 2: Inspeccionar el contenido de los datos

- Una vez que los datos se encuentran abiertos o cargados, lo siguiente que necesitamos es poder **ver su contenido**.
- Existen **diferentes formas** de inspeccionar los datos en R, una vez que estos se encuentran cargados en la sesión.
- Una forma global de inspeccionar los datos, es emplear el comando **dplyr::glimpse**. Este nos entrega el listado de variables, su tipo, y una muestra de los valores contenidos en cada columna de la base de datos.
- Otra forma de inspeccionar los datos, es emplear las funciones de la librería **labelled**. Por ejemplo, empleando el comando **labelled::lookfor** podemos acceder a la meta data de los datos. Una vez aplicado este comando, podemos acceder a la descripción de las variables. Y, para el caso de las variables categóricas, al significado de los valores contenidos en cada columna (e.g., IDCNTRY, COUNTRY).
- También es posible generar un libro de códigos con **labelled**, y guardar este libro de códigos en un archivo **excel**.

pos	variable	label	col_type	levels	value_labels
1	IDSTUD	Identificador del estudiante	dbl	[32]	Argentina
2	IDCLASS	Identificador del aula	dbl	[76]	Brasil
3	ID SCHOOL	Identificador de la escuela	dbl	[170]	Colombia
4	IDCNTRY	Identificador del país	dbl	[183]	Costa Rica
5	IDCNTRY	Identificador del país	dbl	[192]	Cuba
6	IDCNTRY	Identificador del país	dbl	[214]	República Dominicana
7	IDCNTRY	Identificador del país	dbl	[218]	Ecuador
8	IDCNTRY	Identificador del país	dbl	[222]	El Salvador
9	IDCNTRY	Identificador del país	dbl	[320]	Guatemala
10	IDCNTRY	Identificador del país	dbl	[340]	Honduras
11	IDCNTRY	Identificador del país	dbl	[484]	Isla de Pascua
12	IDCNTRY	Identificador del país	dbl	[550]	Nicaragua
13	IDCNTRY	Identificador del país	dbl	[591]	Panamá
14	IDCNTRY	Identificador del país	dbl	[600]	Paraguay
15	IDCNTRY	Identificador del país	dbl	[604]	Perú
16	IDCNTRY	Identificador del país	dbl	[658]	Uruguay
17	IDCNTRY	Identificador del país	chr	[A90]	Argentina
18	IDCNTRY	Identificador del país	chr	[C01]	Colombia
19	IDCNTRY	Identificador del país	chr	[C81]	Costa Rica
20	IDCNTRY	Identificador del país	chr	[CUB]	Cuba
21	5 COUNTRY	Acrónimo del país	chr	[DOM]	Dominican Republic
22	5 COUNTRY	Acrónimo del país	chr	[ECU]	Ecuador
23	5 COUNTRY	Acrónimo del país	chr	[GTM]	Argentina
24	5 COUNTRY	Acrónimo del país	chr	[HND]	Honduras
25	5 COUNTRY	Acrónimo del país	chr	[MEX]	Méjico
26	5 COUNTRY	Acrónimo del país	chr	[NIC]	Nicaragua
27	5 COUNTRY	Acrónimo del país	chr	[PAN]	Panamá
28	5 COUNTRY	Acrónimo del país	chr	[PER]	Perú
29	5 COUNTRY	Acrónimo del país	chr	[PRY]	Paraguay
30	5 COUNTRY	Acrónimo del país	chr	[SLV]	El Salvador
31	5 COUNTRY	Acrónimo del país	chr	[URY]	Uruguay
32	5 COUNTRY	Acrónimo del país	dbl	[1]	Tercer grado
33	5 COUNTRY	Acrónimo del país	dbl	[6]	Sexto grado
34	5 COUNTRY	Acrónimo del país	dbl	[0]	No
35	5 COUNTRY	Acrónimo del país	dbl	[1]	Si
36	5 COUNTRY	Acrónimo del país	dbl	[1]	10 años o menos.
37	6 STRATA	Identificar del estrato del país	dbl	[2]	11 años.
38	7 GRADE	Grado	dbl	[3]	12 años.
39	7 GRADE	Grado	dbl	[4]	13 años.
40	8 QAS	Cuestionario Presente (Sí = 1, No = 0)	dbl	[5]	14 años.
41	8 QAS	Cuestionario Presente (Sí = 1, No = 0)	dbl	[6]	15 años o más.
42	9 WI	Peso total	dbl	[1]	Niña
43	10 WS	Peso al interior de la escuela	dbl	[2]	Niño
44	11 WI	Peso de la escuela (en cada grado)	dbl	[3]	
45	13 E6T01	¿Cuántos años tienes?	dbl	[4]	
46	13 E6T01	¿Cuántos años tienes?	dbl	[5]	
47	13 E6T01	¿Cuántos años tienes?	dbl	[6]	
48	13 E6T01	¿Cuántos años tienes?	dbl	[7]	
49	13 E6T01	¿Cuántos años tienes?	dbl	[8]	
50	13 E6T02	¿Cuántos años tienes?	dbl	[9]	
51	13 E6T02	¿Cuántos años tienes?	dbl	[10]	
52	14 E6T02	¿Eres niña o niño?	dbl	[11]	
53	14 E6T02	¿Eres niña o niño?	dbl	[12]	

Problemas y Soluciones

Variables de diseño

Cuáles son las variables de diseño

Problema 3: **Variables de diseño** (cuáles son las variables de diseño)

- Los estudios de gran escala poseen diferentes **variables de diseño**.

Problema 3: Variables de diseño (cuáles son las variables de diseño)

- Los estudios de gran escala poseen diferentes **variables de diseño**.

```
#--  
# abrir datos  
#--  
  
erce_a6 <- erce::erce_2019_qa6  
  
#--  
# muestra de 10 casos de las variables de diseño  
#--  
  
library(dplyr)  
erce_2019_qa6 %>%  
dplyr::sample_n(10) %>%  
dplyr::select(IDCNTRY, ID SCHOOL, ID STUD, WI, WJ, WT, WS, BRR1, BRR100) %>%  
erce::remove_labels() %>%  
tibble::as_tibble()  
  
  
> library(dplyr)  
> erce::erce_2019_qa6 %>%  
+ dplyr::sample_n(10) %>%  
+ dplyr::select(IDCNTRY, ID SCHOOL, ID STUD, WI, WJ, WT, WS, BRR1, BRR100) %>%  
+ erce::remove_labels() %>%  
+ tibble::as_tibble()  
# A tibble: 10 × 9  
   IDCNTRY ID SCHOOL ID STUD   WI    WJ    WT    WS    BRR1 BRR100  
   <dbl>     <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>  
1      558    1015 10150109  1.24  16.9   21.0   0.357   31.5   31.5  
2      222    1217 12170212  2.01  8.74   17.6   0.177   26.3   8.78  
3      340    1126 11260104  1.11  96.9   108.   0.584   162.   162.  
4      591    1254 12540727    7     1     7     0.105   10.5   10.5  
5      484    1054 10540203  2.06  173.   358.   0.156   537.   179.  
6      320    1065 10650115  1.17  39.2   45.7   0.149   68.6   68.6  
7      222    1236 12360211  2.30  10.6   24.2   0.244   12.1   12.1  
8      604    1182 11820522  5.48  10.8   59.4   0.111   29.7   29.7  
9      604    1237 12370629  6.15  10.4   63.8   0.119   31.9   31.9  
10     222    1079 10790121  6.00  1.37   8.20   0.0825  4.10   4.10
```

Problema 3: Variables de diseño (cuáles son las variables de diseño)

- Los estudios de gran escala poseen diferentes **variables de diseño**.
- Entre las más comunes encontramos a:
 - Las variables de identificación de los **clusters**:
 - El identificador de países (**IDCNTRY**)
 - El identificador de estratos (**STRATA**)
 - El identificador de escuelas (**IDSCHOOL**)
 - El identificador de estudiantes (**IDSTUD**)
 - Las variables de **pesos** de las observaciones:
 - Pesos muestrales totales (**WT**)
 - Pesos muestrales escalados (i.e., *senate weights*) (**WS**)
 - Peso de los estudiantes al interior de las escuelas (**WI**)
 - Pesos de las escuelas (**WJ**)
 - Pesos Replicados (**BRR1-BRR100**)

Problema 3: Variables de diseño (cuáles son las variables de diseño)

- Los estudios de gran escala poseen diferentes **variables de diseño**.
- Entre las más comunes encontramos a:
 - Las variables de identificación de los **clusters**:
 - El identificador de países (**IDCNTRY**)
 - El identificador de estratos (**STRATA**)
 - El identificador de escuelas (**IDSCHOOL**)
 - El identificador de estudiantes (**IDSTUD**)
 - Las variables de **pesos** de las observaciones:
 - Pesos muestrales totales (**WT**)
 - Pesos muestrales escalados (i.e., *senate weights*) (**WS**)
 - Peso de los estudiantes al interior de las escuelas (**WI**)
 - Pesos de las escuelas (**WJ**)
 - Pesos Replicados (**BRR1-BRR100**)

```
#--  
# abrir datos  
#--  
  
erce_a6 <- erce::erce_2019_qa6  
  
#--  
# definir dígitos en consola  
#--  
  
options(digits = 10)  
options(scipen = 999999)  
  
#--  
# muestra de 30 casos de las variables de diseño  
#--  
  
library(dplyr)  
erce:::erce_2019_qa6 %>%  
dplyr::sample_n(20) %>%  
dplyr::select(IDCNTRY, IDSCHOOL, IDSTUD, WI, WJ, WT, WS, BRR1, BRR100) %>%  
erce:::remove_labels() %>%  
tibble::as_tibble() %>%  
knitr::kable(., digits = 2)  
  
> library(dplyr)  
> erce:::erce_2019_qa6 %>%  
+ dplyr::sample_n(20) %>%  
+ dplyr::select(IDCNTRY, IDSCHOOL, IDSTUD, WI, WJ, WT, WS, BRR1, BRR100) %>%  
+ erce:::remove_labels() %>%  
+ tibble::as_tibble() %>%  
+ knitr::kable(., digits = 2)
```

IDCNTRY	IDSCHOOL	IDSTUD	WI	WJ	WT	WS	BRR1	BRR100
484	1120	11200106	1.00	581.92	581.95	0.25	376.20	787.71
76	1021	10210103	4.23	114.49	484.12	0.15	726.18	242.06
320	1012	10120117	2.03	27.70	56.19	0.18	28.09	28.09
214	1053	10530324	3.86	9.74	37.58	0.24	56.36	56.36
218	1083	10830217	4.24	6.38	27.06	0.09	40.60	40.60
340	2183	21830109	2.39	15.45	36.90	0.20	18.45	55.35
76	2024	20240424	4.46	193.25	862.21	0.28	431.10	1293.31
340	2143	21430114	1.10	10.55	11.61	0.06	5.80	5.80
604	1066	10660705	7.00	9.22	64.55	0.12	96.83	96.83
858	1126	11260220	2.16	3.86	8.35	0.17	4.17	4.17
32	1208	12080108	3.00	60.75	182.25	0.22	91.13	91.13
222	1235	12350213	2.46	7.22	17.77	0.18	8.88	26.65
340	1015	10150104	1.52	59.62	90.62	0.49	45.31	45.31
222	1009	10090214	2.36	5.23	12.34	0.12	18.51	6.17
858	1076	10760116	1.03	4.24	4.37	0.09	6.55	2.18
340	1224	12240114	3.18	7.49	23.80	0.13	11.90	11.90
188	1033	10330210	6.29	2.78	17.46	0.26	8.73	8.73
170	1069	10690209	2.20	133.06	292.73	0.35	499.71	499.71
600	1133	11330212	2.00	7.83	15.67	0.15	23.50	7.83
320	1217	12170205	2.12	33.17	70.19	0.23	35.09	105.28

Problemas y soluciones

Variables de diseño

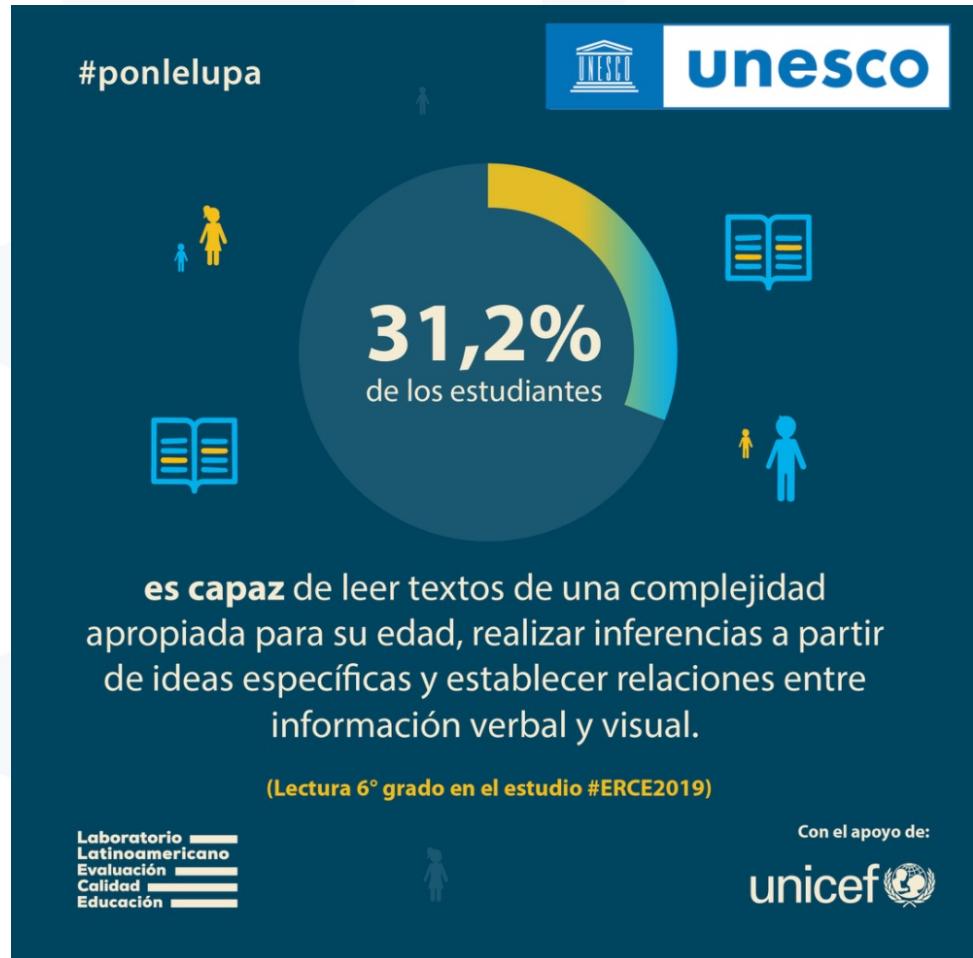
Problema 3: Qué hacer con las variables de diseño, para obtener resultados de la región

Problema 3: Variables de diseño (cómo producir resultados para la región)

- Los **pesos muestrales**, en particular el peso total de las observaciones (**WT**) expande las observaciones al total esperado de la población de interés.
 - Si empleáramos este peso, para tratar de reproducir los resultados regionales sobre los niveles de lectura, nuestro **estimado estaría distorsionado**.
 - Los porcentajes de nivel de lectura, estarían siendo calculados **empleando el total esperado** de la población de cada país. Lo anterior, tiene como consecuencia que, países más grandes como Brasil y México, contribuyen con más observaciones esperadas, en contraste a países con una población de estudiantes de menor tamaño como Uruguay, y Colombia.
- Para garantizar que cada uno de los países contribuya de forma equivalente, requerimos emplear el peso escalado (**WS**). Este peso, suma a un total de mil observaciones de cada país.
- **En resumen**, cuando queremos calcular resultados que involucran más de un país, necesitamos elegir o crear un peso muestral que sea adecuado a nuestros propósitos. En este caso, **WS** cumple este requisito.

Problema 3: Variables de diseño (cómo producir resultados para la región)

- Los **pesos muestrales**, en particular el peso total de las observaciones (**WT**) expande las observaciones al total esperado de la población de interés.
 - Si empleáramos este peso, para tratar de reproducir los resultados regionales sobre los niveles de lectura, nuestro **estimado estaría distorsionado**.
 - Los porcentajes de nivel de lectura, estarían siendo calculados **empleando el total esperado** de la población de cada país. Lo anterior, tiene como consecuencia que, países más grandes como Brasil y México, contribuyen con más observaciones esperadas, en contraste a países con una población de estudiantes de menor tamaño como Uruguay, y Colombia.
- Para garantizar que cada uno de los países contribuya de forma equivalente, requerimos emplear el peso escalado (**WS**). Este peso, suma a un total de mil observaciones de cada país.
- **En resumen**, cuando queremos calcular resultados que involucran más de un país, necesitamos elegir o crear un peso muestral que sea adecuado a nuestros propósitos. En este caso, **WS** cumple este requisito.



Problema 3: Variables de diseño (cómo producir resultados para la región)

- Un segundo aspecto a considerar son los **identificadores de clusters** (e.g., escuelas, estratos y países).
 - Es muy común que las variables de diseño **no sean únicas** entre las bases de datos de países, en los estudios de gran escala. Esto quiere decir que, por ejemplo la escuela **1001** se repite entre todos los países participantes.
 - La repetición de los clusters como los identificadores de escuelas, pueden hacerle creer a los software de turno que **los estudiantes de la escuela 1001 de Colombia, están en la misma escuela 1001 de Ecuador**, y no queremos cometer este tipo de errores.
 - De este modo, si quisiéramos implementar un método de corrección de cálculo de errores como **Taylor Series Linearization**, necesitaremos crear identificadores de cluster que sean únicos entre países.
 - De manera que el software que calcule los resultados no solo corrija el punto estimado de forma adecuada empleando el peso pertinente (**WS**), sino que además considere cómo están anidados las observaciones entre las escuelas, y los estratos de cada país, para obtener tamaños de error razonables.
- **En resumen**, además de elegir o crear pesos adecuados para nuestros propósitos, necesitamos **identificadores de cluster** que sean únicos entre países para este caso.

Problema 3: Variables de diseño (cómo producir resultados para la región)

- Un segundo aspecto a considerar son los **identificadores de clusters** (e.g., escuelas, estratos y países).
 - Es muy común que las variables de diseño **no sean únicas** entre las bases de datos de países, en los estudios de gran escala. Esto quiere decir que, por ejemplo la escuela **1001** se repite entre todos los países participantes.
 - La repetición de los clusters como los identificadores de escuelas, pueden hacerle creer a los software de turno que **los estudiantes de la escuela 1001 de Colombia, están en la misma escuela 1001 de Ecuador**, y no queremos cometer este tipo de errores.
 - De este modo, si quisiéramos implementar un método de corrección de cálculo de errores como **Taylor Series Linearization**, necesitaremos crear identificadores de cluster que sean únicos entre países.
 - De manera que el software que calcule los resultados no solo corrija el punto estimado de forma adecuada empleando el peso pertinente (**WS**), sino que además considere cómo están anidados las observaciones entre las escuelas, y los estratos de cada país, para obtener tamaños de error razonables.
- **En resumen**, además de elegir o crear pesos adecuados para nuestros propósitos, necesitamos **identificadores de cluster** que sean únicos entre países para este caso.

```
#-----  
# abrir datos  
#-----  
  
erce_a6 <- erce::erce_2019_qa6  
  
#-----  
# la escuela 1001 se repite entre todos los países  
#-----  
  
library(dplyr)  
erce_a6 %>%  
erce:::remove_labels() %>%  
dplyr:::select(COUNTRY, IDSCHOOL) %>%  
unique() %>%  
dplyr:::filter(IDSCHOOL == 1001) %>%  
dplyr:::count(COUNTRY, IDSCHOOL) %>%  
knitr:::kable(., digits = 2)
```

COUNTRY	IDSCHOOL	n
ARG	1001	1
BRA	1001	1
COL	1001	1
CRI	1001	1
CUB	1001	1
DOM	1001	1
ECU	1001	1
GTM	1001	1
MEX	1001	1
NIC	1001	1
PAN	1001	1
PER	1001	1
PRY	1001	1
SLV	1001	1
URY	1001	1

Problemas y soluciones

Generación de resultados para la región

Problema 3: Secuencia de códigos para producir un resultado regional

Problema 3: Variables de diseño (cómo producir resultados para la región)

```
# -----  
# nivel de lectura en la region  
# -----  
  
#-----  
# cluster únicos  
#-----  
  
data_lan <- erce::erce_2019_qa6 %>%  
erce::remove_labels() %>%  
mutate(id_s = as.numeric(as.factor(paste0(IDCNTRY, "_", STRATA)))) %>%  
mutate(id_j = as.numeric(as.factor(paste0(IDCNTRY, "_", IDSCHOOL)))) %>%  
mutate(id_i = seq(1:nrow(.)) )  
  
#-----  
# variable dummy para los niveles esperados  
#-----  
  
data_lan <- data_lan %>%  
mutate(all = 1) %>%  
mutate(lan_min_1 = case_when(  
LAN_L1 == 'I' ~ 0,  
LAN_L1 == 'II' ~ 0,  
LAN_L1 == 'III' ~ 1,  
LAN_L1 == 'IV' ~ 1)) %>%  
mutate(lan_min_2 = case_when(  
LAN_L2 == 'I' ~ 0,  
LAN_L2 == 'II' ~ 0,  
LAN_L2 == 'III' ~ 1,  
LAN_L2 == 'IV' ~ 1)) %>%  
mutate(lan_min_3 = case_when(  
LAN_L3 == 'I' ~ 0,  
LAN_L3 == 'II' ~ 0,  
LAN_L3 == 'III' ~ 1,  
LAN_L3 == 'IV' ~ 1)) %>%  
mutate(lan_min_4 = case_when(  
LAN_L4 == 'I' ~ 0,  
LAN_L4 == 'II' ~ 0,  
LAN_L4 == 'III' ~ 1,  
LAN_L4 == 'IV' ~ 1)) %>%  
mutate(lan_min_5 = case_when(  
LAN_L5 == 'I' ~ 0,  
LAN_L5 == 'II' ~ 0,  
LAN_L5 == 'III' ~ 1,  
LAN_L5 == 'IV' ~ 1))
```

Problema 3: Variables de diseño (cómo producir resultados para la región)

```
# -----  
# nivel de lectura en la region  
# -----  
  
#-----  
# cluster únicos  
#-----  
  
data_lan <- erce::erce_2019_qa6 %>%  
erce::remove_labels() %>%  
mutate(id_s = as.numeric(as.factor(paste0(IDCNTRY, " ", STRATA)))) %>%  
mutate(id_j = as.numeric(as.factor(paste0(IDCNTRY, " ", IDSCHOOL)))) %>%  
mutate(id_i = seq(1:nrow(.)) )  
  
#-----  
# variable dummy para los niveles esperados  
#-----  
  
data_lan <- data_lan %>%  
mutate(all = 1) %>%  
mutate(lan_min_1 = case_when(  
LAN_L1 == 'I' ~ 0,  
LAN_L1 == 'II' ~ 0,  
LAN_L1 == 'III' ~ 1,  
LAN_L1 == 'IV' ~ 1)) %>%  
mutate(lan_min_2 = case_when(  
LAN_L2 == 'I' ~ 0,  
LAN_L2 == 'II' ~ 0,  
LAN_L2 == 'III' ~ 1,  
LAN_L2 == 'IV' ~ 1)) %>%  
mutate(lan_min_3 = case_when(  
LAN_L3 == 'I' ~ 0,  
LAN_L3 == 'II' ~ 0,  
LAN_L3 == 'III' ~ 1,  
LAN_L3 == 'IV' ~ 1)) %>%  
mutate(lan_min_4 = case_when(  
LAN_L4 == 'I' ~ 0,  
LAN_L4 == 'II' ~ 0,  
LAN_L4 == 'III' ~ 1,  
LAN_L4 == 'IV' ~ 1)) %>%  
mutate(lan_min_5 = case_when(  
LAN_L5 == 'I' ~ 0,  
LAN_L5 == 'II' ~ 0,  
LAN_L5 == 'III' ~ 1,  
LAN_L5 == 'IV' ~ 1))
```

- Una forma general de llamar a las variables de clustering, es llamarlas id_i, id_j, id_s, y id_k

variables de clustering

id_i = identificador único de estudiantes
id_j = identificador único de escuelas (i.e., primary sampling unit)
id_s = identificador único de estratos
id_k = identificador único de países

- Otro detalle a considerar, es que muchas de las librerías que nos permiten realizar estimaciones a la población, esperan bases de datos sin meta-data. De caso contrario, no reconocen a la base de datos como analizable, y no pueden realizar cálculos. De este modo, en este ejemplo empleamos la función `erce::remove_labels()` para que `library(survey)` no nos genere problemas.
- Además, necesitamos tener una variable **dummy** que identifique a todos los estudiantes que se encuentren sobre el nivel esperado, es decir al menos sobre el nivel III de lectura.
- Como los niveles de desempeño sobre lectura, provienen de los valores plausibles, entonces necesitamos generar 5 variables de este tipo.

Problema 3: Variables de diseño (cómo producir resultados para la región)

```
# -----  
# nivel de lectura en la region  
# -----  
# [... continua código anterior]  
#-----  
# base de datos con diseño  
#-----  
  
# survey method: taylor series linearization  
data_tsl <- survey::svydesign(  
  data = data_lan,  
  weights = ~WS,  
  strata = ~id_s,  
  id = ~id_j,  
  nest = TRUE)  
  
# Opción: corrección a unidad primaria de muestreo que resulte  
# única al estrato  
  
library(survey)  
options(survey.lonely.psu="adjust")
```

Problema 3: Variables de diseño (cómo producir resultados para la región)

```
# -----  
# nivel de lectura en la region  
# -----  
# [...] continua código anterior]  
#-----  
# base de datos con diseño  
#-----  
  
# survey method: taylor series linearization  
data_tsl <- survey::svydesign(  
  data = data_lan,  
  weights = ~WS,  
  strata = ~id_s,  
  id = ~id_j,  
  nest = TRUE)  
  
# Opción: corrección a unidad primaria de muestreo que resulte  
# única al estrato  
  
library(survey)  
options(survey.lonely.psu="adjust")
```

- Primero creamos el objeto **data_tsl**. Este objeto es nuestra base de datos con argumentos necesarios para que la librería **survey** pueda entregarnos estimaciones a la población.
- En esta primera sección del código, estamos indicando el peso que necesitamos emplear (**WS**), además de las variables de cluster que generamos anteriormente (**id_j**, **id_s**).
- Además, necesitamos una función que nos permita combinar los estimados de diferentes **valores plausibles** (Rutkowski et al., 2010). En este caso, estamos empleando una función generada por Thomas Lumley, autor de la librería **survey** (Lumley, 2010), la cual nos permite calcular resultados para cada valor plausible, sin la necesidad de que tengamos que crear una lista de bases de datos.
- En la siguiente lámina, vamos a aplicar esta función **withPV()**, la cual facilitará el proceso de generación de resultados.

Problema 3: Variables de diseño (cómo producir resultados para la región)

```
# -----  
# nivel de lectura en la region  
# -----  
# [... continua código anterior]  
#-----  
# nivel de lectura esperado  
#-----  
  
results <- mitools::withPV(  
  mapping = lan_min ~ lan_min_1 + lan_min_2 + lan_min_3 + lan_min_4 + lan_min_5,  
  data = data_tsl,  
  action=quote(  
    survey::svymean(~lan_min, design=data_tsl)  
  ),  
  rewrite = TRUE  
)  
#-----  
# obtener resultados  
#-----  
  
summary(mitools::MIcombine(results))  
  
  
summary(mitools::MIcombine(results))  
Multiple imputation results:  
  function(mapping, design, action, ...) UseMethod("withPV",design)  
  MIcombine.default(results)  
    results      se      (lower      upper) missInfo  
lan_min 0.3118589523 0.003815336074 0.3041889541 0.3195289505      32 %
```

Problema 3: Variables de diseño (cómo producir resultados para la región)

```
# -----  
# nivel de lectura en la region  
# -----  
# [... continua código anterior]  
#-----  
# nivel de lectura esperado  
#-----  
  
results <- mitools::withPV(  
  mapping = lan_min ~ lan_min_1 + lan_min_2 + lan_min_3 + lan_min_4 + lan_min_5,  
  data = data_tsl,  
  action=quote(  
    survey::svymean(~lan_min, design=data_tsl)  
  ),  
  rewrite = TRUE  
)  
#-----  
# obtener resultados  
#-----  
  
summary(mitoools::MIcombine(results))  
  
  
summary(mitoools::MIcombine(results))  
Multiple imputation results:  
  function(mapping, design, action, ...) UseMethod("withPV",design)  
  MIcombine.default(results)  
    results      se      (lower      upper) missInfo  
  lan_min 0.3118589523 0.003815336074 0.3041889541 0.3195289505      32 %
```

- En la primera sección de este código, estamos aplicando la función **withPV**. En esta función, empleando un esquema de fórmulas, podemos llamar a nuestra variable combinada como quisieramos. En este caso, la estamos llamando **lan_min**.
- Lo importante en la definición de **lan_min**, es que incluyamos los valores plausibles que necesitamos. En este caso, nuestro argumento crítico es:

`mapping = lan_min ~ lan_min_1 + lan_min_2 + lan_min_3 + lan_min_4 + lan_min_5,`
- En **data** = necesitamos incluir nuestro objeto de diseño, en este caso **data_tsl**. Dentro del argumento **action = quote()** vamos a incluir la función de **survey** que nos entregue los resultados que queremos calcular (e.g., svymean, svyglm, svyby). La opción **rewrite = TRUE** asegura que la función sea realizada sobre cada uno de los valores plausibles indicados.
- La aplicación de la función **withPV** entrega una lista de resultados. Esta lista de resultados, necesitamos combinarla empleando las reglas de Rubin (Schafer, 1997). Para estos fines, empleamos la función **mitools::MIcombine(results)** al interior de la función **summary()** de modo de obtener puntos estimados, error del punto estimado e intervalos de confianza.

Problema 3: Variables de diseño (cómo producir resultados para la región)

```
# -----  
# nivel de lectura en la region  
# -----  
# [... continua código anterior]  
#-----  
# tabla de medias  
#-----  
  
estimados <- summary(mitoools::MIcombine(results))  
  
table_read <- estimados %>%  
  tibble::rownames_to_column("lan_min") %>%  
  rename(  
    lan = results,  
    lan_se = se,  
    ll = 4,  
    ul = 5,  
    miss = 6  
  ) %>%  
  mutate(lan = lan*100) %>%  
  mutate(lan_se = lan_se*100) %>%  
  mutate(ll = ll*100) %>%  
  mutate(ul = ul*100)  
  
# -----  
# mostrar tabla  
# -----  
  
options(digits=10)  
options(scipen = 99999)  
  
knitr::kable(table_read, digits = 1)
```

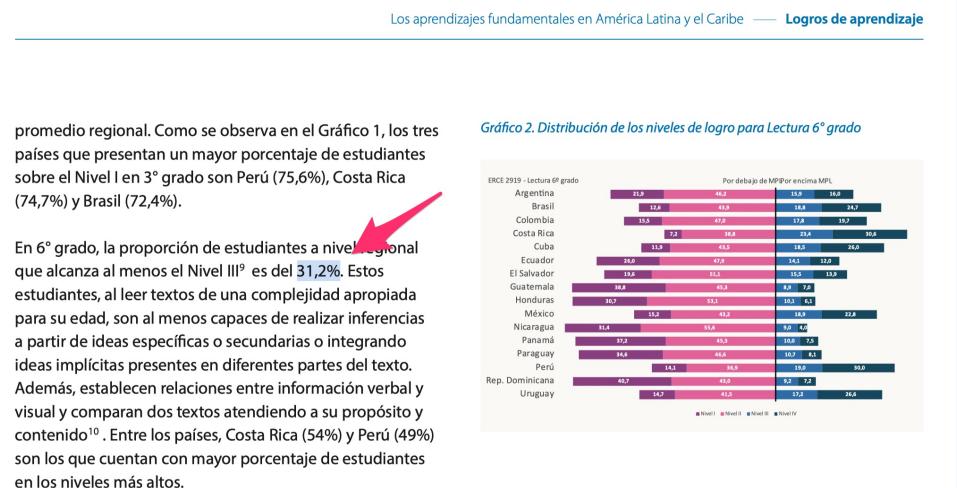
lan_min	lan	lan_se	ll	ul	miss
lan_min	31.2	0.4	30.4	32	32 %

Problema 3: Variables de diseño (cómo producir resultados para la región)

```
# -----  
# nivel de lectura en la region  
# -----  
# [...] continua código anterior]  
#-----  
# tabla de medias  
#-----  
  
estimados <- summary(mitoools::MIcombine(results))  
  
table_read <- estimados %>%  
  tibble::rownames_to_column("lan_min") %>%  
  rename(  
    lan = results,  
    lan_se = se,  
    ll = 4,  
    ul = 5,  
    miss = 6  
  ) %>%  
  mutate(lan = lan*100) %>%  
  mutate(lan_se = lan_se*100) %>%  
  mutate(ll = ll*100) %>%  
  mutate(ul = ul*100)  
  
# -----  
# mostrar tabla  
# -----  
  
options(digits=10)  
options(scipen = 999999)  
  
knitr::kable(table_read, digits = 1)
```

lan_min	lan	lan_se	ll	ul	miss
31.2	0.4	30.4	32	32 %	

- Finalmente, podemos manipular los resultados obtenidos para generar tabla que sea más amigable de leer. Con todo lo anterior, vemos que podemos reproducir los resultados reportados en la serie **#ponlelupa**, y lo que aparece en el informe ejecutivo.



promedio regional. Como se observa en el Gráfico 1, los tres países que presentan un mayor porcentaje de estudiantes sobre el Nivel I en 3º grado son Perú (75,6%), Costa Rica (74,7%) y Brasil (72,4%).

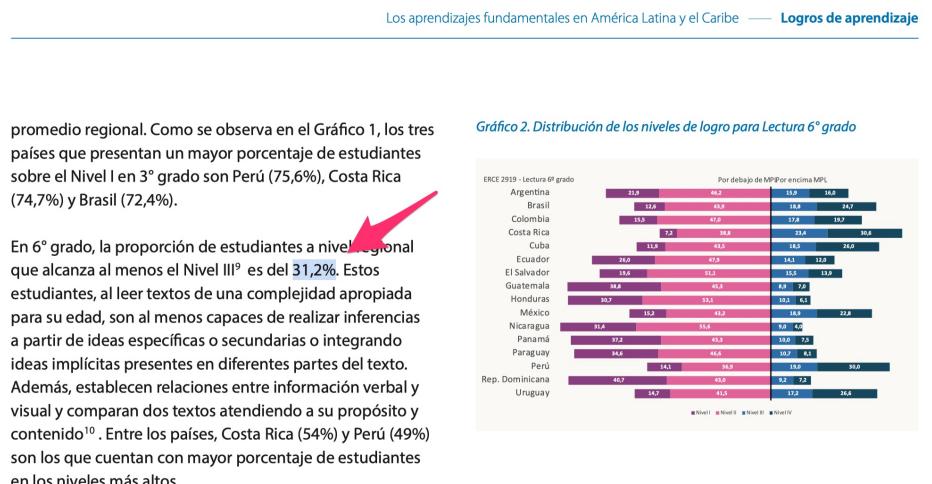
En 6º grado, la proporción de estudiantes a nivel regional que alcanza al menos el Nivel III⁹ es del 31,2%. Estos estudiantes, al leer textos de una complejidad apropiada para su edad, son al menos capaces de realizar inferencias a partir de ideas específicas o secundarias o integrando ideas implícitas presentes en diferentes partes del texto. Además, establecen relaciones entre información verbal y visual y comparan dos textos atendiendo a su propósito y contenido¹⁰. Entre los países, Costa Rica (54%) y Perú (49%) son los que cuentan con mayor porcentaje de estudiantes en los niveles más altos.

Problemas y Soluciones

Generación de resultados para la región

Resumen de todos los pasos realizados para generar un resultado regional

Generación de resultados para la región como desarollo



```
# Resultados generados empleando library(survey) y withPV()
```

```
|lan_min | lan| lan_se| ll| ul|miss |
|-----|---:|----:|---:|---:|---|
|lan_min | 31.2| 0.4| 30.4| 32|32 % |
```

Realizamos una **série de pasos** para poder reproducir la cifra 31.2%, la que nos indica la proporción de estudiantes que alcanza el nivel esperado de competencia de lectura en la región. A la derecha de esta lámina, incluimos un listado de los pasos realizados.

Generación de resultados para la región como desarollo

promedio regional. Como se observa en el Gráfico 1, los tres países que presentan un mayor porcentaje de estudiantes sobre el Nivel I en 3º grado son Perú (75,6%), Costa Rica (74,7%) y Brasil (72,4%).

En 6º grado, la proporción de estudiantes a nivel regional que alcanza al menos el Nivel III⁹ es del 31,2%. Estos estudiantes, al leer textos de una complejidad apropiada para su edad, son al menos capaces de realizar inferencias a partir de ideas específicas o secundarias o integrando ideas implícitas presentes en diferentes partes del texto. Además, establecen relaciones entre información verbal y visual y comparan dos textos atendiendo a su propósito y contenido¹⁰. Entre los países, Costa Rica (54%) y Perú (49%) son los que cuentan con mayor porcentaje de estudiantes en los niveles más altos.

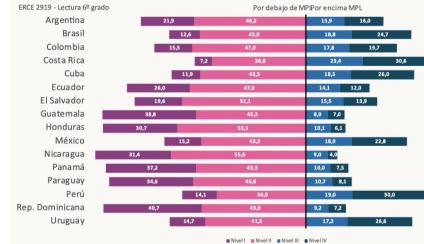
Resultados generados empleando library(survey) y withPV()

```
|lan_min | lan| lan_se| ll| ul|miss |
|-----|---:|-----:|---:|---:|---:|
|lan_min | 31.2| 0.4| 30.4| 32|32 %|
```

Realizamos una **series de pasos** para poder reproducir la cifra 31.2%, la que nos indica la proporción de estudiantes que alcanza el nivel esperado de competencia de lectura en la región. A la derecha de esta lámina, incluimos un listado de los pasos realizados.

Los aprendizajes fundamentales en América Latina y el Caribe — Logros de aprendizaje

Gráfico 2. Distribución de los niveles de logro para Lectura 6º grado



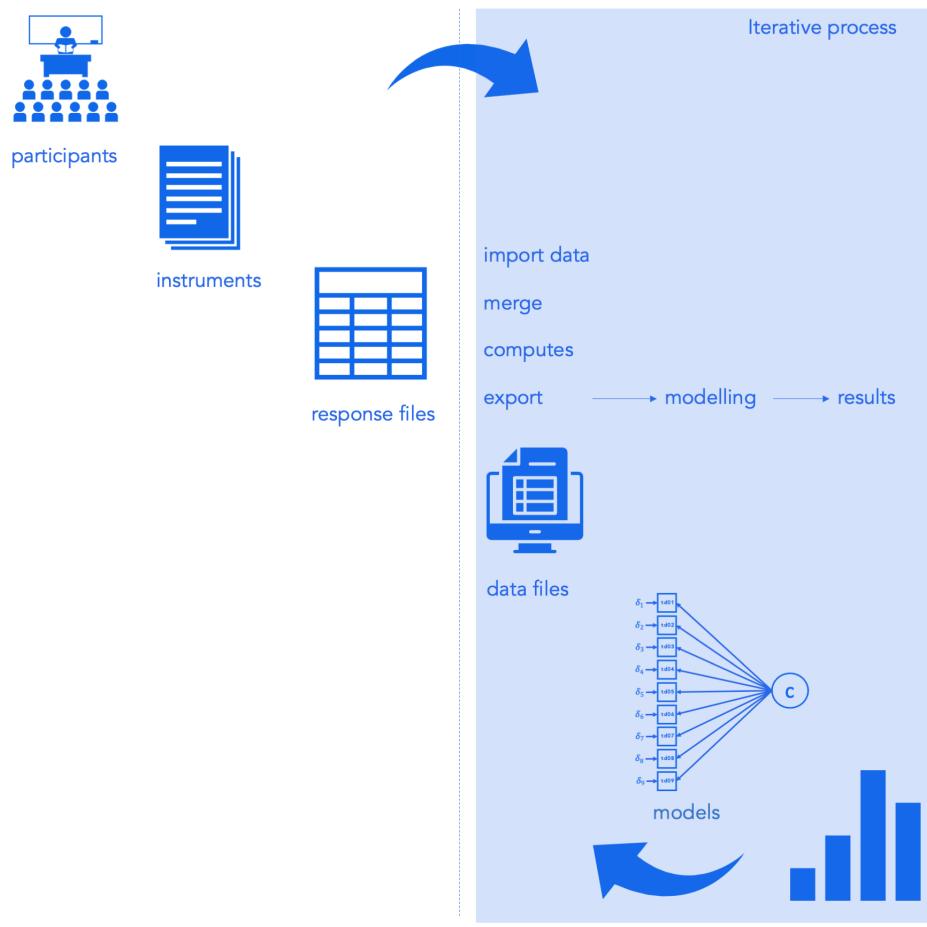
- Abrir los datos ([erce::erce_2019_qa6](#))
- Remover la meta data de los datos ([erce::remove_labels](#))
- La creación de variables clustering que fueran únicas entre países (e.g., `id_i`, `id_j`, `id_s`).
- La creación de variables dummy sobre los niveles de desempeño (e.g., `lan_min_1`, `lan_min_2`, `lan_min_3`, `lan_min_4`, `lan_min_5`).
- La definición de un objeto, que incluya la información del diseño (i.e., `data_tsl`).
- Empleamos a una función que nos permitiera estimar los resultados para cada valor plausible (`withPV`).
- Combinamos los resultados generados, empleando funciones diseñadas para trabajar con imputaciones ([mitools::MIcombine\(results\)](#)).
- Finalmente, **definimos la forma en que se presentan los resultados** en consola, para que esta empleara la misma cantidad de dígitos presentes en el informe ejecutivo, y en el reporte de `#ponlelupa`.

Problemas y Soluciones

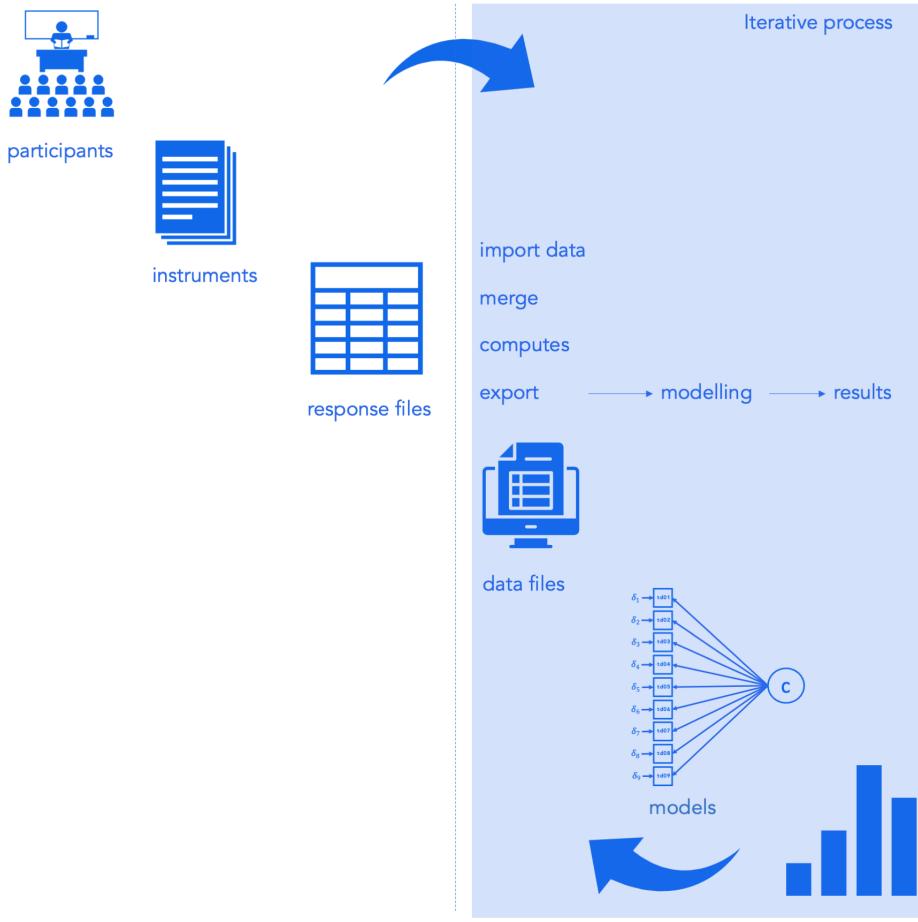
Transferibilidad y Reproducibilidad

Cómo entregar los resultados generados, y como repetir este ejercicio.

Transferibilidad y Reproducibilidad



Transferibilidad y Reproducibilidad



- Para completar el ciclo de **análisis como desarrollo**, tenemos que abordar dos aspectos: la capacidad de **transferir** los resultados generados, y la capacidad de poder **repetir** el proceso.
- El primer aspecto refiere a la **transferibilidad**. Incluiremos como un problema de transferibilidad a la generación de resultados. Para que un resultado generado sea transferible, necesitamos poder **compartir** el resultado con otros usuarios, de alguna manera.
- Una forma de **compartir** el resultado generado, es producir una **tabla en excel**. Esta opción permite que muchos más usuarios tengan acceso a los resultados que hemos producido; y no solo entre usuarios de R.

```
Multiple imputation results:  
withPV.survey.design(mapping = lan_min ~ lan_min_1 + lan_min_2 +  
lan_min_3 + lan_min_4 + lan_min_5, data = data_tsl, action = quote(survey::svymean(~lan_min,  
design = data_tsl)), rewrite = TRUE)  
  MIcombine.default(results)  
    results      se   (lower   upper) missInfo  
lan_min_1  0.311859  0.003815336  0.304189  0.319529     32 %
```

```
#-----  
# nivel de lectura en la region  
#-----  
#--  
# exportar resultados  
#--  
  
table_read %>%  
openxlsx::write.xlsx(.,  
  'tabla_porcentaje_nivel_lecatura.xlsx',  
  overwrite = TRUE)
```

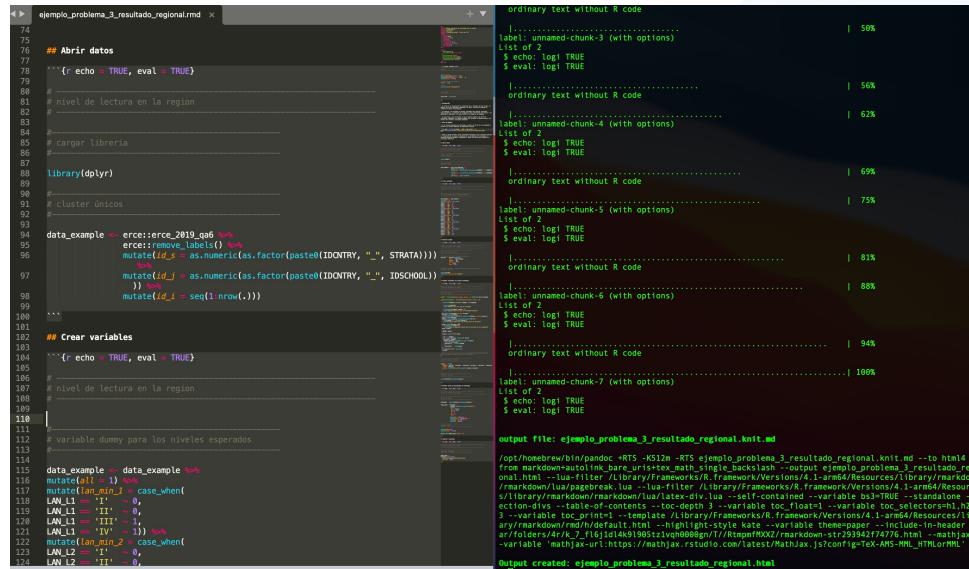
Transferibilidad y Reproducibilidad

	A	B	C	D	E	F	G	H
1	lan_min	lan	lan_se	ll	ul	miss		
2	lan_min	31.18589523	0.381533607	30.41889541	31.95289505	32 %		
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								

- Para completar el ciclo de **análisis como desarrollo**, tenemos que abordar dos aspectos: la capacidad de **transferir** los resultados generados, y la capacidad de poder **repetir** el proceso.
- El primer aspecto refiere a la **transferibilidad**. Incluiremos como un problema de transferibilidad a la generación de resultados. Para que un resultado generado sea transferible, necesitamos poder **compartir** el resultado con otros usuarios, de alguna manera.
- Una forma de **compartir** el resultado generado, es producir una **tabla en excel**. Esta opción permite que muchos más usuarios tengan acceso a los resultados que hemos producido; y no solo entre usuarios de **R**.

```
# nivel de lectura en la region  
#-----  
#--  
# exportar resultados  
#--  
  
table_read %>%  
openxlsx:::write.xlsx(.,  
  'tabla_porcentaje_nivel_lectura.xlsx',  
  overwrite = TRUE)
```

Transferibilidad y Reproducibilidad



```
ejemplo_problema_3_resultadoRegional.Rmd
74
75
76 ## Abrir datos
77
78 ##(r echo = TRUE, eval = TRUE)
79
80 # nivel de lectura en la region
81 #
82 #
83 #
84 # cargar libreria
85 #
86 library(dplyr)
87 #
88 # cluster unicos
89 #
90 #
91 data_example <- erce::erce_2019_ga6 %>%
92   mutate(id_s = as.numeric(as.factor(paste0(IDONTRY, "_", IDSCHOOL)))
93     )
94   mutate(id_d = as.numeric(as.factor(paste0(IDONTRY, "_", STRATA))))
95   )
96   mutate(id_s = seq(1:nrow(.)))
97
98 ##
99
100 ##
101 ## Crear variables
102
103 ##(r echo = TRUE, eval = TRUE)
104
105 #
106 # nivel de lectura en la region
107 #
108 #
109 #
110 #
111 #
112 # variable dummy para los niveles esperados
113 #
114 #
115 data_example <- data_example %>%
116   mutate(id_l = 1) %>
117   case_when(
118     LAN_L1 == "I" ~ 0,
119     LAN_L1 == "II" ~ 0,
120     LAN_L1 == "III" ~ 1,
121     LAN_L1 == "IV" ~ 1,
122     LAN_L1 == "V" ~ 1) %>
123   mutate(id_mn_2 = case_when(
124     LAN_L2 == "I" ~ 0,
125     LAN_L2 == "II" ~ 0,
```

```
ordinary text without R code
|:label: unnamed-chunk-3 (with options)
|:List of 2
|:  $ echo: logi TRUE
|:  $ eval: logi TRUE
|: ordinary text without R code
|:label: unnamed-chunk-4 (with options)
|:List of 2
|:  $ echo: logi TRUE
|:  $ eval: logi TRUE
|: ordinary text without R code
|:label: unnamed-chunk-5 (with options)
|:List of 2
|:  $ echo: logi TRUE
|:  $ eval: logi TRUE
|: ordinary text without R code
|:label: unnamed-chunk-6 (with options)
|:List of 2
|:  $ echo: logi TRUE
|:  $ eval: logi TRUE
|: ordinary text without R code
|:label: unnamed-chunk-7 (with options)
|:List of 2
|:  $ echo: logi TRUE
|:  $ eval: logi TRUE
|: ordinary text without R code
output file: ejemplo_problema_3_resultadoRegional.html
/opt/homebrew/bin/pandoc -RTS -K128 -RTS ejemplo_problema_3_resultadoRegional.knit.md --to html4
from markdown+autolink_bare_uris+tex_math_single_backslash -output ejemplo_problema_3_resultadoRegional.html
--filter /Library/Frameworks/R.framework/Versions/4.1-arm64/Resources/library/rmarkdown/rmd/convert/markdown.R --filter /Library/Frameworks/R.framework/Versions/4.1-arm64/Resources/library/rmarkdown/lua/latex-div.lua --self-contained --variable bi3=TRUE --standalone --economics --toc-level=3 --variable toc_float=TRUE --variable section_depth=3 --variable page_title="Análisis de la transferibilidad y reproducibilidad del problema 3" --variable header-includes=<!--include<br></head><head><link href="https://mathjax.rstudio.com/latest/MathJax.js?config=TeX-AMS-MML_HTMLorMML" rel="stylesheet"></head>--> --variable theme=paper --include-in-header /Library/Frameworks/R.framework/Versions/4.1-arm64/Resources/library/rmarkdown/rmd/default.html --highlight-style kate --variable mathjax_url="https://mathjax.rstudio.com/latest/MathJax.js?config=TeX-AMS-MML_HTMLorMML"
Output created: ejemplo_problema_3_resultadoRegional.html
```

Nota: imagen de como se ve un código reproducible (Xie, 2013), en formato **RMarkdown**. A la izquierda se encuentra el código escrito en Sublime Text, mientras que a la derecha se encuentra un screenshot de la terminal de **R** mostrando los logs de la ejecución exitosa del código.

Este código lo pueden bajar desde el siguiente link:

https://github.com/dacarras/erce_2022_lsa/blob/main/02_ejemplos/02_desarrollo_codigos.rmd

- Finalmente, todos los pasos revisados hasta ahora, los podemos juntar en un solo **código**.
- Desde la apertura de los datos, la creación de variables de clustering, la creación de variables, la estimación de resultados con valores plausibles, la combinación de las estimaciones realizadas, y la definición de la forma de presentación de resultados, la podemos agrupar en un mismo documento **accionario**.

Transferibilidad y Reproducibilidad



The screenshot shows a web browser window with the URL [\(Personal\)/_erce_2019/40_taller_jlisa/ ejemplo_problema_3_resultado Regional.html](#). The page title is "Definir forma de presentación de resultados". On the left, there's a sidebar with navigation links: Introducción, Cifra de Ejemplo, Abrir datos, Crear variables, Declarar diseño, Estimar resultados con valores plausibles, Definir forma de presentación de resultados (which is selected), and Exportar resultados. The main content area contains R code:

```
# -----
# nivel de lectura en la region
# -----
#-----
# tabla de medias
#-----
estimados <- summary(mitools::Mimcombine(results))

## Multiple imputation results:
##   function(mapping, design, action, ...) UseMethod("withPV",design)
##   Mimcombine.default(results)
##   results      se (lower upper) missInfo
##   lan_min     0.31 0.0038   0.3   0.32    32 %

table_read <- estimados %>
  tibble::rownames_to_column("lan_min") %>%
  rename(
    lan = results,
    lan_se = se,
    ll = 4,
    ul = 5,
    miss = 6
  ) %>
  mutate(lan = lan*100) %>
  mutate(ll = lan_se*100) %>
  mutate(ll = ll*100) %>
  mutate(ul = ul*100)

# -----
# mostrar tabla
```

Nota: imagen del output html generado con el código escrito en **RMarkdown**. Ver [02_desarrollo_codigo.rmd](#).

- Finalmente, todos los pasos revisados hasta ahora, los podemos juntar en un solo **código**.
- Desde la apertura de los datos, la creación de variables de clustering, la creación de variables, la estimación de resultados con valores plausibles, la combinación de las estimaciones realizadas, y la definición de la forma de presentación de resultados, la podemos agrupar en un mismo documento **accionable**.
- Una ventaja de implementar esta forma de trabajo, es que el código generado, no solo nos entrega la tabla de resultados en excel, la cual es **transferible**. Sino que además, el código escrito que produce los resultados es **reproducible**. Es decir, que nos permite generar los resultados nuevamente, y plasma los resultados encontrados en esta tarea.

Transferibilidad y Reproducibilidad

```
# -----
# nivel de lectura en la region
# -----

#-----
# tabla de medias
#-----



estimados <- summary(mitools::Mimcombine(results))

## Multiple imputation results:
##   function(mapping, design, action, ...) UseMethod("withPV",design)
##   Mimcombine.default(results)
##   results      se (lower upper) missInfo
##   lan_min     0.31 0.0038   0.3  0.32    32 %



table_read <- estimados %>
  tibble::rownames_to_column("lan_min") %>%
  rename(
    lan = results,
    lan_se = se,
    ll = 4,
    ul = 5,
    miss = 6
  ) %>%
  mutate(lan = lan*100) %>%
  mutate(ll = lan_se*100) %>%
  mutate(ll = ll*100) %>%
  mutate(ul = ul*100)

# -----



# mostrar tabla
```

Nota: imagen del output html generado con el código escrito en **RMarkdown**. Ver [02_desarollo_codigo.rmd](#).

- Finalmente, todos los pasos revisados hasta ahora, los podemos juntar en un solo **código**.
- Desde la apertura de los datos, la creación de variables de clustering, la creación de variables, la estimación de resultados con valores plausibles, la combinación de las estimaciones realizadas, y la definición de la forma de presentación de resultados, la podemos agrupar en un mismo documento **accionable**.
- Una ventaja de implementar esta forma de trabajo, es que el código generado, no solo nos entrega la tabla de resultados en excel, la cual es **transferible**. Sino que además, el código escrito que produce los resultados es **reproducible**. Es decir, que nos permite generar los resultados nuevamente, y plasma los resultados encontrados en esta tarea.
- Este tipo de código facilita que **reutilicemos** el código empleado, para resolver problemas similares. Y además implementa el proceso **iterativo**. Es decir, que podemos repetir la secuencia de pasos hasta llegar al producto final, incorporando mejoras en cada iteración.

Transferibilidad y Reproducibilidad

The screenshot shows a web browser window with the URL [\(Personal\)/_erce_2019/40_taller_llsa/ejemplo_problema_3_resultado Regional.html](#). The page title is "Definir forma de presentación de resultados". On the left, there is a sidebar with navigation links: Introducción, Cifra de Ejemplo, Abrir datos, Crear variables, Declarar diseño, Estimar resultados con valores plausibles, Definir forma de presentación de resultados, and Exportar resultados. The main content area contains R code:

```
# -----
# nivel de lectura en la region
# -----
#-----
# tabla de medias
#-----
estimados <- summary(mitools::Mimcombine(results))

## Multiple imputation results:
##   function(mapping, design, action, ...) UseMethod("withPV",design)
##   Mimcombine.default(results)
##   results      se (lower upper) missInfo
##   lan_min     0.31 0.0038  0.3  0.32    32 %

table_read <- estimados %>
  tibble::rownames_to_column("lan_min") %>%
  rename(
    lan = results,
    lan_se = se,
    ll = 4,
    ul = 5,
    miss = 6
  ) %>
  mutate(lan = lan*100) %>
  mutate(ll = lan_se*100) %>
  mutate(ll = ll*100) %>
  mutate(ul = ul*100)

# -----
# mostrar tabla
```

Nota: imagen del output html generado con el código escrito en **RMarkdown**. Ver [02_desarrollo_codigo.rmd](#).

- Finalmente, todos los pasos revisados hasta ahora, los podemos juntar en un solo **código**.
- Desde la apertura de los datos, la creación de variables de clustering, la creación de variables, la estimación de resultados con valores plausibles, la combinación de las estimaciones realizadas, y la definición de la forma de presentación de resultados, la podemos agrupar en un mismo documento **accionable**.
- Una ventaja de implementar esta forma de trabajo, es que el código generado, no solo nos entrega la tabla de resultados en excel, la cual es **transferible**. Sino que además, el código escrito que produce los resultados es **reproducible**. Es decir, que nos permite generar los resultados nuevamente, y plasma los resultados encontrados en esta tarea.
- Este tipo de código facilita que **reutilicemos** el código empleado, para resolver problemas similares. Y además implementa el proceso **iterativo**. Es decir, que podemos repetir el proceso de secuencia de pasos, hasta llegar al producto final, incorporando mejoras en cada iteración.
- En este taller, invitamos a los presentes a probar esta forma de trabajo, y esperamos que resulte ventajoso para la **producción de resultados como desarollo**.

Muchas gracias!

Referencias

- Lumley, T. (2010). Complex Surveys: A Guide to Analysis Using R. Wiley.
- Parker, H. (2017). Opinionated analysis development (pp. 1–13).
<https://doi.org/10.7287/peerj.preprints.3210>
- Rutkowski, L., Gonzalez, E., Joncas, M., & von Davier, M. (2010). International Large-Scale Assessment Data: Issues in Secondary Analysis and Reporting. *Educational Researcher*, 39(2), 142–151.
<https://doi.org/10.3102/0013189X10363170>
- Schafer, J. L. (1997). Analysis of Incomplete Multivariate Data. In We (Vol. 141, Issue 7). Chapman & Hall/CRC.
- Xie, Y. (2013). Dynamic Documents with R and knitr. In Dynamic Documents with R and knitr. Chapman and Hall/CRC.
<http://www.crcpress.com/product/isbn/9781482203530>