

R - Taller DESUC

Manejo de base de datos

DESUC

21/01/2019

Motivación

¿Por qué?

- Trabajamos con datos
- Invertir tiempo en lo que importa

Motivación

¿Por qué?

- integración de principio a fin
- automatización, reportes y reproducibilidad
- gráficos
- paquetes y comunidad

R: Lenguaje vs Software

¿Qué es lo que lo hace tan intimidante?

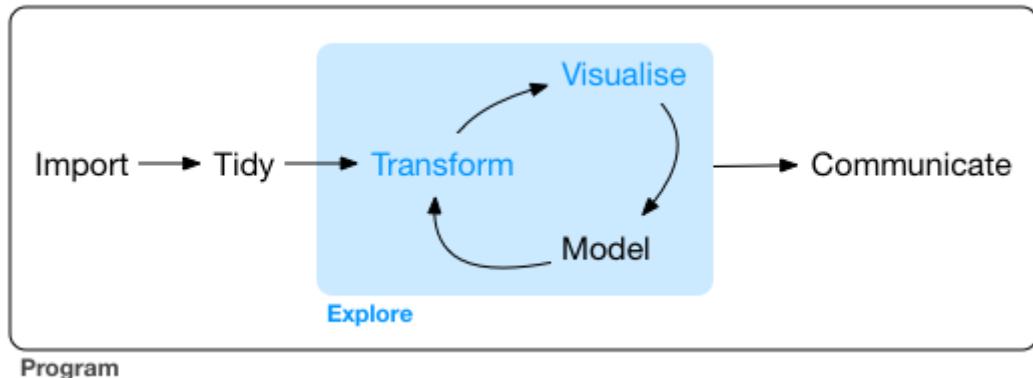
análisis <—> programación

Definiciones

- tidyverse

The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures.

```
include_graphics('images/data-science-explore.png')
```



Definiciones

- tidy data

Tidy data sets are arranged such that each variable is a column and each observation (or case) is a row

```

not_tidy <- tibble(Año = c(2017, 2018, 2019),
                    Cualitativo = c(5, 4, 1),
                    Cuantitativo = c(15, 16, 1),
                    Organizacional = c(4, 3, 0))
not_tidy

## # A tibble: 3 x 4
##   Año Cualitativo Cuantitativo Organizacional
##   <dbl>      <dbl>        <dbl>          <dbl>
## 1 2017         5            15             4
## 2 2018         4            16             3
## 3 2019         1            1              0

```

Definiciones

- tidy data

Tidy data sets are arranged such that each variable is a column and each observation (or case) is a row

```

tidy <- not_tidy %>%
  gather('Unidad', 'n', -Año)

tidy %>% head()

```

```

## # A tibble: 6 x 3
##   Año Unidad     n
##   <dbl> <chr>    <dbl>
## 1 2017 Cualitativo  5
## 2 2018 Cualitativo  4
## 3 2019 Cualitativo  1
## 4 2017 Cuantitativo 15
## 5 2018 Cuantitativo 16
## 6 2019 Cuantitativo  1

```

Estructura de datos estandarizada para poder utilizar las herramientas de *tidyverse*.

Objetivo

Utilizar R y tidyverse para el manejo de base de datos:

- Lectura de bases de datos
- Manipulación de variables
- Uso y manejo de etiquetas

Ejercicio

Uso de *directorio* y *matrícula* de alumnos de 2018, provista por el MINEDUC.

Queremos obtener la siguiente información según establecimiento:

- Categorizar establecimientos según áreas del Gran Santiago
- Establecimientos según si son CH y/o TP
- Establecimientos con enseñanza básica y/o media
- Número de hombres y mujeres por establecimiento
- Edad promedio de sus estudiantes NNA

Paquetes

```
library(readxl) # Lectura de archivos excel
library(haven) # Lectura de SPSS o Stata
library(tidyverse) # Básico: dplyr, tidyr, purrr, stringr, forcats
library(janitor) # Arreglo de nombres y tablas

library(sjlabelled) # Manejo de etiquetas
library(sjmisc) # Funciones de ayuda

library(knitr) # Creación de documentos mediante Rmarkdown
```

Lectura de datos

Lectura de datos: Excel

Datos de comunas que componen el Gran Santiago

- xlsx: readxl::read_excel

```
df_gran_santiago <- read_excel('..../data/gran_santiago.xlsx',
                                sheet = 1,
                                trim_ws = TRUE)
```

Tips:

- Designar rangos específicos para leer información (range =)
- Saltar algunas filas en blanco (skip =)

Lectura de datos: Excel

Explorar base de datos:

¿Cómo se ve la base de datos que acabo de leer?

```
glimpse(df_gran_santiago)
```

```
## Observations: 34
## Variables: 3
## $ comuna  <dbl> 13101, 13118, 13120, 13123, 13129, 13130, 13104, 13107...
## $ ncomuna <chr> "Santiago", "Macul", "Ñuñoa", "Providencia", "San Joaq...
## $ zonas   <chr> "Centro", "Centro", "Centro", "Centro", "Centro", "Cen...
```

Lectura de datos: Bases en texto

Matrícula de alumnos 2018.

- CSV: readr::read_csv o readr::read_csv2.

Diferencia entre funciones es el delimitador de columnas. Para datos en Chile, en general, se usaría read_csv2.

```
df_mat_18 <- read_csv2('..../data/20181005_Matrícula_unica_2018_20180430_PUBL.gs.csv.zip')
```

Tips:

- Se puede leer csv comprimidos en zip
- Se puede configurar los parámetros si se quisiese (read_delim)

Lectura de datos: Bases en texto

¿Cuáles son el nombre de las variables de la base de matrícula?

```
names(df_mat_18)
```

```
## [1] "AGNO"           "RBD"            "DGV_RBD"        "NOM_RBD"
## [5] "COD_REG_RBD"    "COD_PRO_RBD"    "COD_COM_RBD"    "NOM_COM_RBD"
## [9] "COD_DEPROV_RBD" "NOM_DEPROV_RBD" "COD_DEPE"        "COD_DEPE2"
## [13] "RURAL_RBD"      "ESTADO_ESTAB"   "COD_ENSE"       "COD_ENSE2"
## [17] "COD_ENSE3"       "COD_GRADO"      "COD_GRADO2"     "LET_CUR"
## [21] "COD_JOR"         "COD_TIP_CUR"    "COD.Des_CUR"   "MRUN"
## [25] "GEN_ALU"         "FEC_NAC_ALU"   "EDAD_ALU"       "COD_REG_ALU"
## [29] "COD_COM_ALU"    "NOM_COM_ALU"   "COD_SEC"        "COD_ESPE"
## [33] "COD_RAMA"        "ENS"
```

¿Cuál es el tamaño de la base?

```
dim(df_mat_18)
```

```
## [1] 354062      34
```

Lectura de datos: SPSS

Directorio de establecimientos

- SPSS: haven::read_sav

```
df_dir_18 <- read_sav('..../data/20181005_Directorio_Oficial_EE_2018_20180430_PUBL.rm.sav')
```

Tips:

- Lee etiquetas de preguntas y variables.

Lectura de datos: SPSS

¿Cuál eses son las etiquetas de las variables en la base?

```
sjlabelled::get_label(df_dir_18)
```

```
##          AGNO           RBD          DGV_RBD
##          "Año"          ""          ""
##          NOM_RBD        MRUN        RUT_SOSTENEDOR
##          "Nombre de RBD"  ""          ""
##          P_JURIDICA     COD_REG_RBD  COD_PRO_RBD
##          ""              ""          ""
##          COD_COM_RBD    NOM_COM_RBD  COD_DEPROV_RBD
##          ""              ""          ""
##          NOM_DEPROV_RBD COD_DEPE     COD_DEPE2
##          ""  "Código de dependencia"  ""
##          RURAL_RBD      LATITUD     LONGITUD
##          "Ruralidad"     ""          ""
##          ENS_01           ENS_02     ENS_03
##          ""              ""          ""
##          ENS_04           ENS_05     ENS_06
##          ""              ""          ""
##          ENS_07           ENS_08     ENS_09
##          ""              ""          ""
##          MATRICULA       ESTADO_ESTAB ORI_RELIGIOSA
```

```

##          ""
##      ORI_OTRO_GLOSA      PAGO_MATRICULA      PAGO_MENSUAL
##          ""                  ""                  ""
##      ens_basica      ""
##          ""

```

Lectura de datos: SPSS

¿Y las etiquetas de los niveles?

```

df_dir_18 %>%
  select(COD_DEPE) %>%
  sjlabelled::get_labels()

## $COD_DEPE
## [1] "Corporación Municipal"
## [2] "Municipal DAEM"
## [3] "Particular subvencionado"
## [4] "Participar pagado"
## [5] "Corporación de Administración Delegada"
## [6] "Servicio Local de Educación"

```

Análisis

Análisis: herramienta

Para el manejo de base de datos, usaremos `dplyr`. En resumen:

- `filter`: filtra casos según una condición
- `select`: selecciona variables según nombre
- `mutate`: crea variables
- `group_by`: agrupa casos según una variable
- `summarise`: crea resumen de variables

Todos estos verbos se enlazan con la pipa `%>%`

Análisis: la pipa

Permite poner comandos según secuencia de ejecución.

Hace la lectura de código más sencilla y clara.

```

exp(sqrt(9))

## [1] 20.08554

9 %>%
  sqrt(.) %>%
  exp(.)

## [1] 20.08554

x <- 9
x <- sqrt(x)
exp(x)

## [1] 20.08554

```

Análisis: tip

Hay muy buenos *cheatsheets* donde se pueden revisar diversas funciones existentes.

```
knitr:::include_graphics('dplyr.pdf')
```

Data Transformation with dplyr :: CHEAT SHEET



dplyr functions work with pipes and expect **tidy data**. In tidy data:



Each variable is in its own column



Each observation, or case, is in its own row



`x %>% f(y)` becomes `f(x, y)`

Summarise Cases

These apply **summary functions** to columns to create a new table. Summary functions take vectors as input and return one value (see back).

summary function →

`summarise_(.data, ...)`
Compute table of summaries. Also `summarise_()`.

`count(x, ..., wt = NULL, sort = FALSE)`
Count number of rows in each group defined by the variables in ... Also `tally()`.
`count(iris, Species)`

VARIATIONS

`summarise_all()` - Apply funs to every column.
`summarise_at()` - Apply funs to specific columns.
`summarise_if()` - Apply funs to all cols of one type.

Group Cases

Use `group_by()` to create a "grouped" copy of a table. dplyr functions will manipulate each "group" separately and then combine the results.

`mtcars %>% group_by(cyl) %>% summarise(avg = mean(mpg))`

`group_by(data, ..., add = FALSE)`
Returns copy of table grouped by ...
`g_iris <- group_by(iris, Species)`

RStudio® is a trademark of RStudio, Inc. • CC BY RStudio • info@rstudio.com • 844-448-1212 • rstudio.com • Learn more with `browseVignettes(package = c("dplyr", "tibble"))` • dplyr 0.5.0 • tibble 1.2.0 • Updated: 2017-01

Manipulate Cases

EXTRACT CASES

Row functions return a subset of rows as a new table. Use a variant that ends in _ for non-standard evaluation friendly code.

`filter(.data, ...)` Extract rows that meet logical criteria. Also `filter_()`. `filter(iris, Sepal.Length > 7)`

`distinct(.data, ..., keep_all = FALSE)` Remove rows with duplicate values. Also `distinct_()`.
`distinct(iris, Species)`

`sample_frac(tbl, size = 1, replace = FALSE, weight = NULL, env = parent.frame())` Randomly select fraction of rows.
`sample_frac(iris, 0.5, replace = TRUE)`

`sample_n(tbl, size, replace = FALSE, weight = NULL, env = parent.frame())` Randomly select size rows. `sample_n(iris, 10, replace = TRUE)`

`slice(.data, ...)` Select rows by position. Also `slice_()`. `slice(iris, 10:15)`

`top_n(x, n, wt)` Select and order top n entries (by group if grouped data). `top_n(iris, 5, Sepal.Width)`

Logical and boolean operators to use with filter()

`<` `<=` `is.na()` `%in%` `|` `xor()`

See `?base::logic` and `?Comparison` for help.

ARRANGE CASES

`arrange(.data, ...)`
Order rows by values of a column (low to high), use with `desc()` to order from high to low.
`arrange(mtcars, mpg)`
`arrange(mtcars, desc(mpg))`

ADD CASES

`add_row(.data, ..., .before = NULL, .after = NULL)`
Add one or more rows to a table.
`add_row(faithful, eruptions = 1, waiting = 1)`

Column functions return a set of columns as a new table. Use a variant that ends in _ for non-standard evaluation friendly code.

`select(.data, ...)`
Extract columns by name. Also `select_if()`.
`select(iris, Sepal.Length, Species)`

Use these helpers with select(),

e.g. `select(iris, starts_with("Sepo"))`
`contains(match)` `num_range(prefix, range)` ;, e.g. `mpg:cyl`
`ends_with(match)` `one_of(...)` ;, e.g. `-Species`
`matches(match)` `starts_with(match)`

MAKE NEW VARIABLES

These apply **vectorized functions** to columns. Vectorized funs take vectors as input and return vectors of the same length as output (see back).

vectorized function →

`mutate(.data, ...)`
Compute new column(s).
`mutate(mtcars, gpm = 1/mpg)`

`transmute(.data, ...)`
Compute new column(s), drop others.
`transmute(mtcars, gpm = 1/mpg)`

`mutate_all(tbl, funs, ...)` Apply funs to every column. Use with `fun()`.
`mutate_all(faithful, funs(log10), log2(.))`

`mutate_at(tbl, .cols, funs, ...)` Apply funs to specific columns. Use with `fun()`, `vars()` and the helper functions for select().
`mutate_at(iris, vars(-Species), funs(log(.)))`

`mutate_if(tbl, .predicate, .funs, ...)` Apply funs to all columns of one type. Use with `fun()`.
`mutate_if(iris, is.numeric, funs(log(.)))`

`add_column(.data, ..., .before = NULL, .after = NULL)` Add new column(s).
`add_column(mtcars, new = 1:32)`

`rename(.data, ...)` Rename columns.
`rename(iris, Length = Sepal.Length)`



Análisis Gran Santiago: casos

Utilizaré base de df_gran_santiago porque es más pequeña que las otras dos.

¿Cómo se ve la base de datos que acabo de leer?

- Ver primeras tres filas `head()`.
- Ver últimas filas con `tail()`

```
df_gran_santiago %>% head(3)
```

```
## # A tibble: 3 x 3
##   comuna ncomuna zonas
##   <dbl> <chr>    <chr>
## 1 13101 Santiago Centro
## 2 13118 Macul     Centro
## 3 13120 Ñuñoa    Centro

df_gran_santiago %>% tail(3)
```

```
## # A tibble: 3 x 3
##   comuna ncomuna     zonas
##   <dbl> <chr>      <chr>
## 1 13131 San Ramón  Sur
## 2 13201 Puente Alto  Sur
## 3 13401 San Bernardo  Sur
```

Análisis Gran Santiago: filtrar casos

Usar función `filter()`.

```
df_gran_santiago %>%
  filter(zonas == 'Norte')
```

```
## # A tibble: 6 x 3
##   comuna ncomuna     zonas
##   <dbl> <chr>      <chr>
## 1 13104 Conchalií    Norte
## 2 13107 Huechuraba  Norte
## 3 13108 Independencia Norte
## 4 13125 Quilicura    Norte
## 5 13127 Recoleta    Norte
## 6 13128 Renca        Norte
```

Análisis Gran Santiago: ordenar casos

Ordenar la base en base al código comunal `arrange(comuna)`.

```
df_gran_santiago %>% arrange(comuna) %>% head(3)
```

```
## # A tibble: 3 x 3
##   comuna ncomuna     zonas
##   <dbl> <chr>      <chr>
## 1 13101 Santiago  Centro
## 2 13102 Cerrillos  Poniente
## 3 13103 Cerro Navia Poniente
```

Análisis Gran Santiago: grupos

¿Cuál es la cantidad de comunas por zona? Agrupar por zona `group_by(zonas)` y luego calcular dato por grupo `summarise()`.

```
df_gran_santiago %>%
  group_by(zonas) %>%
  summarise(comunas_n = n())
```

```
## # A tibble: 5 x 2
##   zonas     comunas_n
##   <chr>       <int>
## 1 Centro        6
## 2 Norte         6
## 3 Oriente      5
## 4 Poniente     9
## 5 Sur          8
```

Análisis: Matrícula

¿Cuál es el número de alumnos según dependencia?

```
df_mat_18 %>%
  group_by(COD_DEPE2) %>%
  summarise(alumnos_n = n())
```

```
## # A tibble: 4 x 2
##   COD_DEPE2 alumnos_n
##       <dbl>     <int>
## 1           1     90117
## 2           2    123270
## 3           3    129913
## 4           4     10762
```

Análisis: Matrícula

¿Podemos agregar la proporción?

```
df_mat_18 %>%
  group_by(COD_DEPE2) %>%
  summarise(alumnos_n = n()) %>%
  mutate(alumnos_prop = alumnos_n / sum(alumnos_n))
```

```
## # A tibble: 4 x 3
##   COD_DEPE2 alumnos_n alumnos_prop
##       <dbl>     <int>        <dbl>
## 1           1     90117      0.255
## 2           2    123270      0.348
## 3           3    129913      0.367
## 4           4     10762      0.0304
```

Análisis: Matrícula

¿Y si queremos saber el número de alumnos por sexo?

- Alternativa *no-tidy*

```
df_mat_18 %>%
  group_by(COD_DEPE2) %>%
  summarise(alumnos_mas_n = sum(GEN_ALU == 1),
            alumnos_fem_n = sum(GEN_ALU == 2)) %>%
  mutate_at(vars(starts_with('alumnos')), funs(prop = ./sum(.)))
```

```
## # A tibble: 4 x 5
##   COD_DEPE2 alumnos_mas_n alumnos_fem_n alumnos_mas_n_pr~ alumnos_fem_n_pr~
##       <dbl>        <int>        <int>        <dbl>        <dbl>
## 1           1        49402        40715      0.274      0.235
## 2           2        60457        62813      0.335      0.362
## 3           3        66029        63884      0.366      0.368
## 4           4        4626         6136      0.0256     0.0354
```

Análisis: Matrícula

¿Y si queremos saber el número de alumnos por sexo? ¡Agrupar por la variable de sexo!

- Alternativa *tidy*

```

df_mat_18 %>%
  group_by(COD_DEPE2, GEN_ALU) %>%
  summarise(alumnos_n = n()) %>%
  mutate(alumnos_prop = alumnos_n / sum(alumnos_n))

## # A tibble: 8 x 4
## # Groups:   COD_DEPE2 [4]
##   COD_DEPE2 GEN_ALU alumnos_n alumnos_prop
##   <dbl>     <dbl>     <int>        <dbl>
## 1 1         1         1    49402      0.548
## 2 2         1         2    40715      0.452
## 3 3         2         1    60457      0.490
## 4 4         2         2    62813      0.510
## 5 5         3         1    66029      0.508
## 6 6         3         2    63884      0.492
## 7 7         4         1    4626       0.430
## 8 8         4         2    6136       0.570

```

Análisis: Matrícula

Hay funciones que facilitan la tarea

```
sjmisc::frq(df_mat_18, COD_DEPE2)
```

```

##
## # COD_DEPE2 <numeric>
## # total N=354062  valid N=354062  mean=2.17  sd=0.84
##
##   val     frq raw.prc valid.prc cum.prc
##   1 90117    25.45    25.45   25.45
##   2 123270   34.82    34.82   60.27
##   3 129913   36.69    36.69   96.96
##   4 10762    3.04     3.04   100.00
##   <NA>      0     0.00      NA      NA

```

Análisis: Matrícula

Hay funciones que facilitan la tarea

```
df_mat_18 %>%
  group_by(GEN_ALU) %>%
  sjmisc::frq(COD_DEPE)
```

```

##
## Grouped by:
## GEN_ALU: 1
##
## # COD_DEPE <numeric>
## # total N=180514  valid N=180514  mean=2.99  sd=1.09
##
##   val     frq raw.prc valid.prc cum.prc
##   1 26961    14.94    14.94   14.94
##   2 22441    12.43    12.43   27.37
##   3 60457    33.49    33.49   60.86
##   4 66029    36.58    36.58   97.44

```

```

##      5 4626    2.56      2.56 100.00
## <NA>     0    0.00       NA       NA
##
## Grouped by:
## GEN_ALU: 2
##
## # COD_DEPE <numeric>
## # total N=173548  valid N=173548  mean=3.06  sd=1.09
##
##   val   frq raw.prc valid.prc cum.prc
##   1 25650    14.78    14.78   14.78
##   2 15065     8.68     8.68   23.46
##   3 62813    36.19    36.19   59.65
##   4 63884    36.81    36.81   96.46
##   5 6136     3.54     3.54  100.00
## <NA>     0    0.00       NA       NA

```

Etiquetas

Agregar etiquetas

Agregar etiquetas a variable y niveles

```

df_mat_18 <- df_mat_18 %>%
  mutate(COD_DEPE2 = labelled(COD_DEPE2,
    labels = c('Municipal' = 1,
              'Particular Subvencionado' = 2,
              'Particular Pagado' = 3,
              'Administración Delegada' = 4,
              'Servicio Local de Educación' = 5),
    label = 'Código de Dependencia del Establecimiento (agrupado)')

frq(df_mat_18, COD_DEPE2)

##
## # Código de Dependencia del Establecimiento (agrupado) (COD_DEPE2) <numeric>
## # total N=354062  valid N=354062  mean=2.17  sd=0.84
##
##   val           label   frq raw.prc valid.prc cum.prc
##   1      Municipal  90117   25.45    25.45   25.45
##   2  Particular Subvencionado 123270   34.82    34.82   60.27
##   3      Particular Pagado 129913   36.69    36.69   96.96
##   4  Administración Delegada 10762    3.04     3.04  100.00
##   5 Servicio Local de Educación     0    0.00     0.00  100.00
##   NA                    NA     0    0.00       NA       NA

```

Agregar etiquetas: multiples variables

```

etiquetas <- c('AGNO' = 'Año escolar',
              'RBD' = 'Rol base de datos del establecimiento',
              'NOM_RBD' = 'Nombre del Establecimiento')

set_label(df_mat_18[, names(etiquetas)]) <- etiquetas

```

```

df_mat_18 %>%
  select(1:2, NOM_RBD) %>%
  get_label()

##                                     AGNO
##                               "Año escolar"
##                               RBD
## "Rol base de datos del establecimiento"
##                               NOM_RBD
## "Nombre del Establecimiento"

```

Agregar niveles: multiples variables

```

niveles <- c('110' = 'Enseñanza Básica',
            '310' = 'Enseñanza Media H-C niños y jóvenes',
            '410' = 'Enseñanza Media T-P Comercial Niños y Jóvenes',
            '510' = 'Enseñanza Media T-P Industrial Niños y Jóvenes')

df_dir_18 <- set_labels(df_dir_18,
                         ENS_01, ENS_02,
                         labels = niveles)

df_dir_18 %>%
  filter(ENS_01 %in% names(niveles)) %>%
  frq(ENS_02)

##
## # ENS_02 <numeric>
## # total N=444  valid N=444  mean=237.38  sd=193.94
##
##   val                                label frq raw.prc valid.prc
##   0                                    0 155  34.91    34.91
##   110                                 Enseñanza Básica  0  0.00    0.00
##   165                                 165   6  1.35    1.35
##   212                                 212   4  0.90    0.90
##   214                                 214   7  1.58    1.58
##   310                                 Enseñanza Media H-C niños y jóvenes 165  37.16    37.16
##   363                                 363   14  3.15    3.15
##   410                                 Enseñanza Media T-P Comercial Niños y Jóvenes 39  8.78    8.78
##   510                                 Enseñanza Media T-P Industrial Niños y Jóvenes 35  7.88    7.88
##   610                                 610   17  3.83    3.83
##   710                                 710   1  0.23    0.23
##   910                                 910   1  0.23    0.23
##   NA                                  NA   0  0.00    NA
##
##   cum.prc
##   34.91
##   34.91
##   36.26
##   37.16
##   38.74
##   75.90
##   79.05
##   87.84
##   95.72

```

```

##    99.55
##    99.77
##   100.00
##      NA

```

Manejo de bases

Análisis: Código de enseñanza

¿Cuántas escuelas tienen Enseñanza Media H-C ENS == 310 en el directorio de establecimientos?

```

df_dir_18 %>%
  select(RBD, NOM_RBD, ENS_01:ENS_04)

```

```

## # A tibble: 4,148 x 6
##       RBD NOM_RBD          ENS_01 ENS_02 ENS_03 ENS_04
##   <dbl> <chr>          <dbl>  <dbl>  <dbl>  <dbl>
## 1 1646 COLEGIO CRISTIANO SAN LUCAS      0      0      0      0
## 2 6880 COLEGIO ANGLO AMERICAN INTERNATIONAL      0      0      0      0
## 3 8485 LICEO INSTITUTO NACIONAL      110     310      0      0
## 4 8487 LICEO JAVIERA CARRERA      110     310      0      0
## 5 8488 LICEO ISAURA DINATOR DE GUZMAN      110     310      0      0
## 6 8489 LICEO BICENTENARIO TERESA PRATS      110     310      0      0
## 7 8490 LICEO NRO 2 MIGUEL LUIS AMUNATEGUI      110     310      0      0
## 8 8491 LICEO DE APLICACION RECTOR JORGE E SC~      110     165     310     363
## 9 8492 LICEO MANUEL BARROS BORGONO      110     310      0      0
## 10 8494 LICEO A-1 VALENTIN LETELIER      310      0      0      0
## # ... with 4,138 more rows

```

Análisis: Código de enseñanza

Tenemos que cambiar la forma de esta base de datos para responder esta pregunta.

```

df_dir_18_long <- df_dir_18 %>%
  gather('variable', 'ENS', starts_with('ENS'))

df_dir_18_long %>%
  arrange(RBD) %>%
  select(RBD, NOM_RBD, ENS) %>%
  head(7)

```

```

## # A tibble: 7 x 3
##       RBD NOM_RBD          ENS
##   <dbl> <chr>          <dbl>
## 1 1646 COLEGIO CRISTIANO SAN LUCAS      0
## 2 1646 COLEGIO CRISTIANO SAN LUCAS      0
## 3 1646 COLEGIO CRISTIANO SAN LUCAS      0
## 4 1646 COLEGIO CRISTIANO SAN LUCAS      0
## 5 1646 COLEGIO CRISTIANO SAN LUCAS      0
## 6 1646 COLEGIO CRISTIANO SAN LUCAS      0
## 7 1646 COLEGIO CRISTIANO SAN LUCAS      0

```

Falta eliminar los datos *vacios*, que en este caso tienen código de enseñanza igual a cero.

Análisis: Código de enseñanza

Depurar base con función filter().

```
df_dir_18_long <- df_dir_18_long %>%
  filter(ENS != 0)

df_dir_18_long %>%
  arrange(RBD) %>%
  select(RBD, NOM_RBD, ENS)

## # A tibble: 8,198 x 3
##       RBD NOM_RBD          ENS
##   <dbl> <chr>        <dbl>
## 1 8485 LICEO INSTITUTO NACIONAL 110
## 2 8485 LICEO INSTITUTO NACIONAL 310
## 3 8485 LICEO INSTITUTO NACIONAL 1
## 4 8487 LICEO JAVIERA CARRERA 110
## 5 8487 LICEO JAVIERA CARRERA 310
## 6 8487 LICEO JAVIERA CARRERA 1
## 7 8488 LICEO ISAURA DINATOR DE GUZMAN 110
## 8 8488 LICEO ISAURA DINATOR DE GUZMAN 310
## 9 8488 LICEO ISAURA DINATOR DE GUZMAN 1
## 10 8489 LICEO BICENTENARIO TERESA PRATS 110
## # ... with 8,188 more rows
```

Análisis: Código de enseñanza

Contar número de casos según variable ENS.

```
df_dir_18_long %>%
  group_by(ENS) %>%
  summarise(ENS_n = n()) %>%
  filter(ENS %in% c(110, 310))

## # A tibble: 2 x 2
##       ENS ENS_n
##   <dbl> <int>
## 1    110 1893
## 2    310  991
```

Análisis: Código de enseñanza

¿Cuántos establecimientos tienen básica (110) y C-H (310)?

```
df_dir_bas_y_ch <- df_dir_18_long %>%
  group_by(RBD, NOM_RBD) %>%
  summarise(basica_y_ch = sum(ENS %in% c(110, 310)) == 2) %>%
  ungroup()

df_dir_bas_y_ch %>%
  slice(c(1, 7:8))

## # A tibble: 3 x 3
##       RBD NOM_RBD          basica_y_ch
##   <dbl> <chr>            <lgl>
## 1 8485 LICEO INSTITUTO NACIONAL TRUE
```

```

## 2 8492 LICEO MANUEL BARROS BORGONO TRUE
## 3 8494 LICEO A-1 VALENTIN LETELIER FALSE
sum(df_dir_bas_y_ch$basica_y_ch) / nrow(df_dir_bas_y_ch)

## [1] 0.2982869

```

Unión de bases

Análisis: Directorio y matrícula

¿Cuántos alumnos hay por nivel de enseñanza? ¿Cuál es la edad promedio que tienen?

```

df_rbd_edad_sexo <- df_mat_18 %>%
  group_by(RBD, COD_ENSE) %>%
  summarise(alumnos_n = n(),
            edad_mean = mean(EDAD_ALU),
            sexo_prop = mean(GEN_ALU == 1))

df_rbd_edad_sexo %>%
  head(4)

## # A tibble: 4 x 5
## # Groups:   RBD [2]
##       RBD COD_ENSE alumnos_n  edad_mean sexo_prop
##     <dbl>    <dbl>    <int>      <dbl>      <dbl>
## 1  8485        110     1517      12.7     1
## 2  8485        310     2764      15.6     1
## 3  8487        110      935      12.7  0.00107
## 4  8487        310     2027      15.6  0.000493

```

Análisis: Directorio y matrícula

Pegar datos de alumnos a directorio de establecimientos por nivel de enseñanza.

Cambiaré el nombre de una variable con función `rename()`.

```

df_rbd_edad_sexo <- df_rbd_edad_sexo %>%
  rename(ENS = COD_ENSE)

df_dir_18_long_mat <- left_join(df_dir_18_long,
                                   df_rbd_edad_sexo,
                                   by = c('RBD', 'ENS'))

names(df_dir_18_long_mat)

## [1] "AGNO"          "RBD"           "DGV_RBD"        "NOM_RBD"
## [5] "MRUN"           "RUT_SOSTENEDOR" "P_JURIDICA"     "COD_REG_RBD"
## [9] "COD_PRO_RBD"    "COD_COM_RBD"   "NOM_COM_RBD"   "COD_DEPROV_RBD"
## [13] "NOM_DEPROV_RBD" "COD_DEPE"       "COD_DEPE2"      "RURAL_RBD"
## [17] "LATITUD"        "LONGITUD"      "MATRICULA"     "ESTADO_ESTAB"
## [21] "ORI_RELIGIOSA"  "ORI_OTRO_GLOSA" "PAGO_MATRICULA" "PAGO_MENSUAL"
## [25] "variable"       "ENS"           "alumnos_n"      "edad_mean"
## [29] "sexo_prop"

```

Análisis: Directorio y matrícula

¿Cuántos alumnos tienen básica (110) y C-H (310)?

```
df_dir_18_long_mat %>%
  group_by(ENS) %>%
  summarise(alumnos_n = sum(alumnos_n)) %>%
  filter(ENS %in% c(110, 310))
```

```
## # A tibble: 2 x 2
##       ENS   alumnos_n
##     <dbl>     <int>
## 1    110      NA
## 2    310      NA
```

R es sensible con valores perdidos NA.

```
df_dir_18_long_mat %>%
  group_by(ENS) %>%
  summarise(alumnos_n = sum(alumnos_n, na.rm = TRUE)) %>%
  filter(ENS %in% c(110, 310))
```

```
## # A tibble: 2 x 2
##       ENS   alumnos_n
##     <dbl>     <int>
## 1    110    182551
## 2    310    92962
```

Futuro: integración entre paquetes

Posición de escuelas en RM

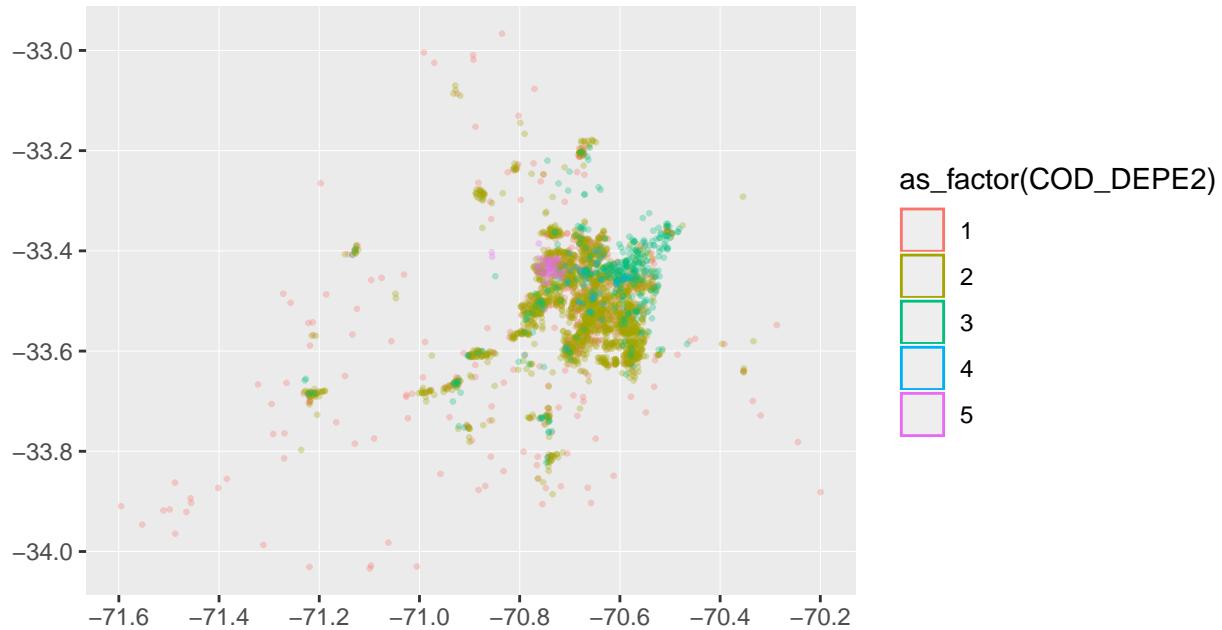
Construir una base de datos con información geográfica a partir de longitud y latitud presente en la base.

```
df_dir_18_geo <- df_dir_18 %>%
  filter(!is.na(LONGITUD)) %>%
  st_as_sf(coords = c("LONGITUD", "LATITUD"))
```

Posición de escuelas en RM

Ver la ubicación geográfica

```
df_dir_18_geo %>%
  ggplot(aes(colour = as_factor(COD_DEPE2))) +
  geom_sf(alpha = 0.3, size = 0.5)
```



Faltan las etiquetas

Posición de escuelas en RM

Agregar etiquetas a variable COD_DEPE2. La tenemos en la base de datos.

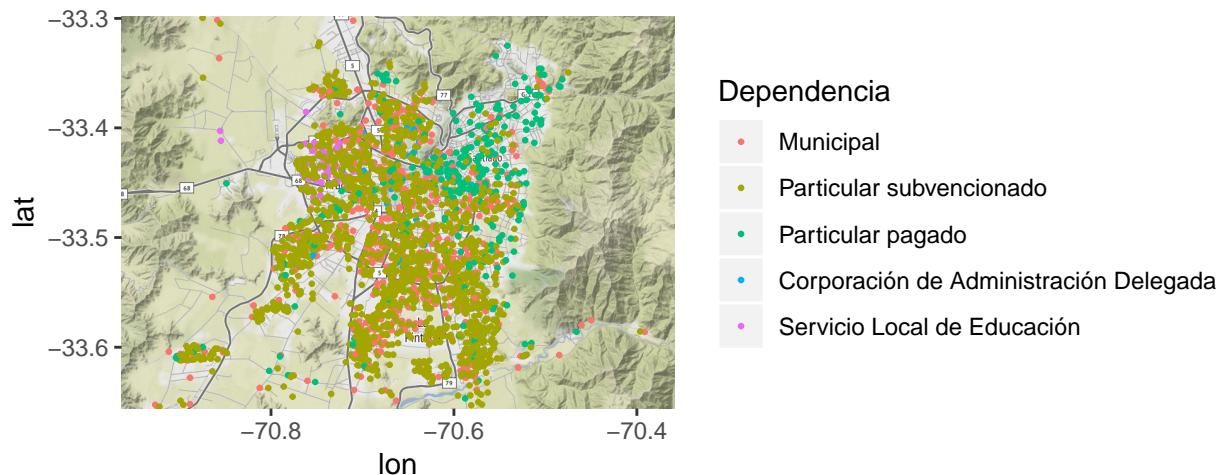
```
gg <- df_dir_18_geo %>%
  ggplot(aes(colour = as_label(COD_DEPE2))) +
  geom_sf(alpha = 0.3, size = 0.5) +
  scale_color_discrete(name = 'Dependencia')
```

Posición de escuelas en RM: ggmap

```
library(ggmap)

gran_santiago <- c(left = -70.9634, bottom = -33.6558, right = -70.3592, top = -33.2984)
mapa_base <- get_stamenmap(gran_santiago, zoom = 11)

ggmap(mapa_base) +
  geom_point(data = df_dir_18, aes(x = LONGITUD, y = LATITUD, colour = as_label(COD_DEPE2)), size = 0.5)
```



Posición de escuelas en RM: leaflet

```
library(leaflet)

leaflet() %>% # leaflet works with the pipe operator
  addTiles() %>% # setup the default OpenStreetMap map tiles
  addMarkers(lng = df_dir_18$LONGITUD, lat = df_dir_18$LATITUD, popup = df_dir_18$NOM_RBD)
```