CLASE 06 - MANIPULAR Y TRANSFORMAR DATOS.

Diplomado en Análisis de Datos con R e Investigación reproducible para Biociencias.

Dr. José Gallardo Matus | https://genomics.pucv.cl/

Pontificia Universidad Católica de Valparaíso

17 September 2022

PLAN DE LA CLASE

1.- Introducción

- ¿Para qué manipular datos?
- Diferencia entre Tidy and messy data.
- Paquete tidyr.
- Operador pipe (Tuberías).
- Paquete dplyr.

2). Práctica con R y Rstudio cloud.

- Realizar manipulación de datos con tidyr y dplyr.
- Realizar gráficas avanzadas con ggplot2.

MANIPULACIÓN DE DATOS

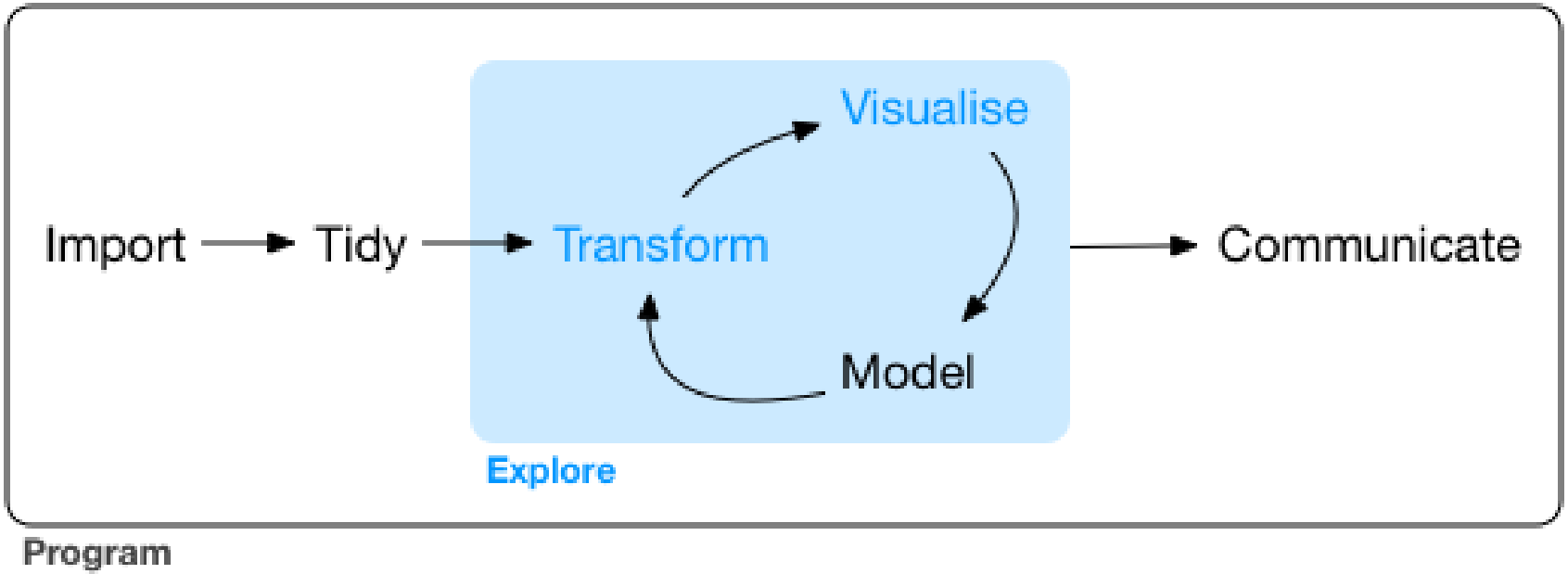
¿Para qué manipular datos?

- Para hacer datos más legibles y organizados.
- Para dar formato adecuado previo a visualización y análisis estadístico.

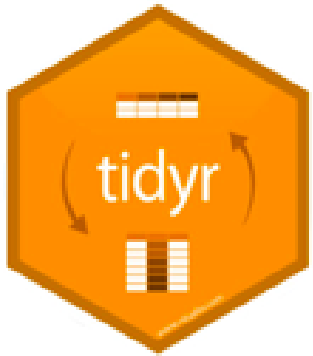
Ejemplos de tareas comunes durante esta etapa:

- Filtrar datos por categorías.
- Remover o imputar datos faltantes.
- Agrupar datos por algún criterio.
- Seleccionar y calcular estadísticos.
- Generar variables derivadas a partir de variables existentes.
- Transformar variables.

ETAPAS DEL ANÁLISIS DE DATOS



PAQUETES CLAVE



DATOS TIDY - MESSY

Tidy data (datos ordenados)

- Cada columna es una variable.
- Cada fila es una observación.
- Cada celda es un simple dato o valor.

Messy data (desordenados)

- Cualquier conjunto de datos que no cumple alguno de estos criterios.

EJEMPLO DATOS MESSY

¿Por qué son messy?

Variable	Replica	Especie A	Especie B	Especie C
peso	1	174	NA	135
peso	2	155	103	138
peso	3	131	138	135
parásitos	1	25	8	5
parásitos	2	12	3	8
parásitos	3	4	11	NA

EJEMPLO DATOS TIDY

¿Por qué son tidy?

Pez	Especie	Sexo	Peso	Parásitos
1	A	Hembra	174	25
2	A	Hembra	155	12
3	A	Hembra	131	4
4	B	Macho	NA	8
5	B	Macho	103	3
6	B	Hembra	138	11
7	C	Hembra	135	5
8	C	Macho	138	8
9	C	Hembra	135	NA

EL OPERADOR PIPE: %>%.

En programación **pipe** es una técnica que permite pasar información de un proceso o programa a otro por etapas.

Evita pipe cuando: a) Deseas manipular varios objetos a la vez. b) Un paso intermedio genera un objeto que luego deseas analizar separadamente.

PAQUETE TIDYR: FUNCIONES CLAVE

gather(): Colapsa múltiples columnas para crear tidy data.
spread(): Separa una columna en múltiples columnas.

País	Ene.	May.	Sep.
Chile	22	13	12
Ecuador	8,8	8,0	8,0

gather(...)

spread(...)

País	Mes	TºC
Chile	Ene.	22
Ecuador	Ene.	8,8
Chile	May.	13
Ecuador	May.	8,0
Chile	Sep.	12
Ecuador	Sep.	8,0

gather("Mes","TºC",2:4)

spread("Mes","TºC")

PAQUETE DPLYR: FUNCIONES BÁSICAS

select(): Permite extraer o seleccionar variables/columnas específicas de un data.frame.

filter(): Para filtrar desde una tabla de datos un subconjunto de filas. Ej. solo un nivel de un factor, observaciones que cumplen algún criterio (ej. > 20).

mutate(): Permite calcular/generar nuevas variables "derivadas". Útil para calcular proporciones, tasas.

arrange(): Permite ordenar la base de datos según una variable de forma ascendente o descendente.

PAQUETE DPLYR: SELECT()

Datos %>% select(Mes, TºC)

País	Mes	TºC
Chile	Ene.	22
Ecuador	Ene.	8,8
Chile	May.	13
Ecuador	May.	8,0
Chile	Sep.	12
Ecuador	Sep.	8,0

Mes	TºC
Ene.	22
Ene.	8,8
May.	13
May.	8,0
Sep.	12
Sep.	8,0

PAQUETE DPLYR: FILTER()

```
Datos %>% filter(País="Chile")
```

País	Mes	TºC
Chile	Ene.	22
Ecuador	Ene.	8,8
Chile	May.	13
Ecuador	May.	8,0
Chile	Sep.	12
Ecuador	Sep.	8,0

Pais	Mes	TºC
Chile	Ene.	22
Chile	May.	13
Chile	Sep.	12

PAQUETE DPLYR: MUTATE()

Datos %>% mutate(Fahrenheit=(TºC*1.8)+32)

País	Mes	TºC
Chile	Ene.	22
Ecuador	Ene.	8,8
Chile	May.	13
Ecuador	May.	8,0
Chile	Sep.	12
Ecuador	Sep.	8,0

País	Mes	TºC	Fahrenheit
Chile	Ene.	22	71,6
Ecuador	Ene.	8,8	47,8
Chile	May.	13	55,4
Ecuador	May.	8,0	46,4
Chile	Sep.	12	53,6
Ecuador	Sep.	8,0	46,4

PAQUETE DPLYR: FUNCIONES AVANZADAS

group_by(): Permite agrupar filas con base a los niveles de alguna variable o factor.

summarize(): Permite obtener medidas resumen de las variables.

inner_join(): Permite unir 2 data.frames con las filas que tienen información en ambas tablas usando una variable índice.

full_join(): Permite unir 2 data.frames con todas las filas.

left_join(): Permite unir 2 data.frames con todas las filas de la primera tabla, si no hay datos conmpleta con NA.

right_join(): Permite unir 2 data.frames con todas las filas de la segunda tabla, si no hay datos conmpleta con NA.

PAQUETE DPLYR: GROUP_BY + SUMMARIZE()

```
Datos %>% grup_by(País) %>%
          summarize(n=n(),
                    promedio=mean(TºC))
```

País	Mes	TºC
Chile	Ene.	22
Ecuador	Ene.	8,8
Chile	May.	13
Ecuador	May.	8,0
Chile	Sep.	12
Ecuador	Sep.	8,0

Bahía	Año	TSM
Chile	Ene.	22
Chile	May.	13
Chile	Sep.	12
Ecuador	Ene.	8,8
Ecuador	May.	8,0
Ecuador	Sep.	8,0

Bahía	n	Promedio
Chile	3	15,7
Ecuador	3	8,3

PAQUETE DPLYR: JOIN DATA FRAME

tb1

Index	TºC
Bogota	20
Quito	10
Santiago	18

tb2

Index	mm
Bogota	800
Lima	5
Santiago	80

left_join(tb1, tb2, by = "Index")

Index	TºC	mm
Bogota	20	800
Quito	10	*NA*
Santiago	18	80

full_join(tb1, tb2, by = "Index")

Index	TºC	mm
Bogota	20	8
Lima	NA	5
Quito	10	NA
Santiago	18	10

RESUMEN DE LA CLASE

- Diferenciamos datos ordenados (Tidy) y desordenados (Messy).
- Manipulamos datos con tidyr y dplyr.
- Utilizamos tuberías o pipe **%>%**.
- Hicimos gráfico ggplot2 usando datos transformados y variables derivadas.