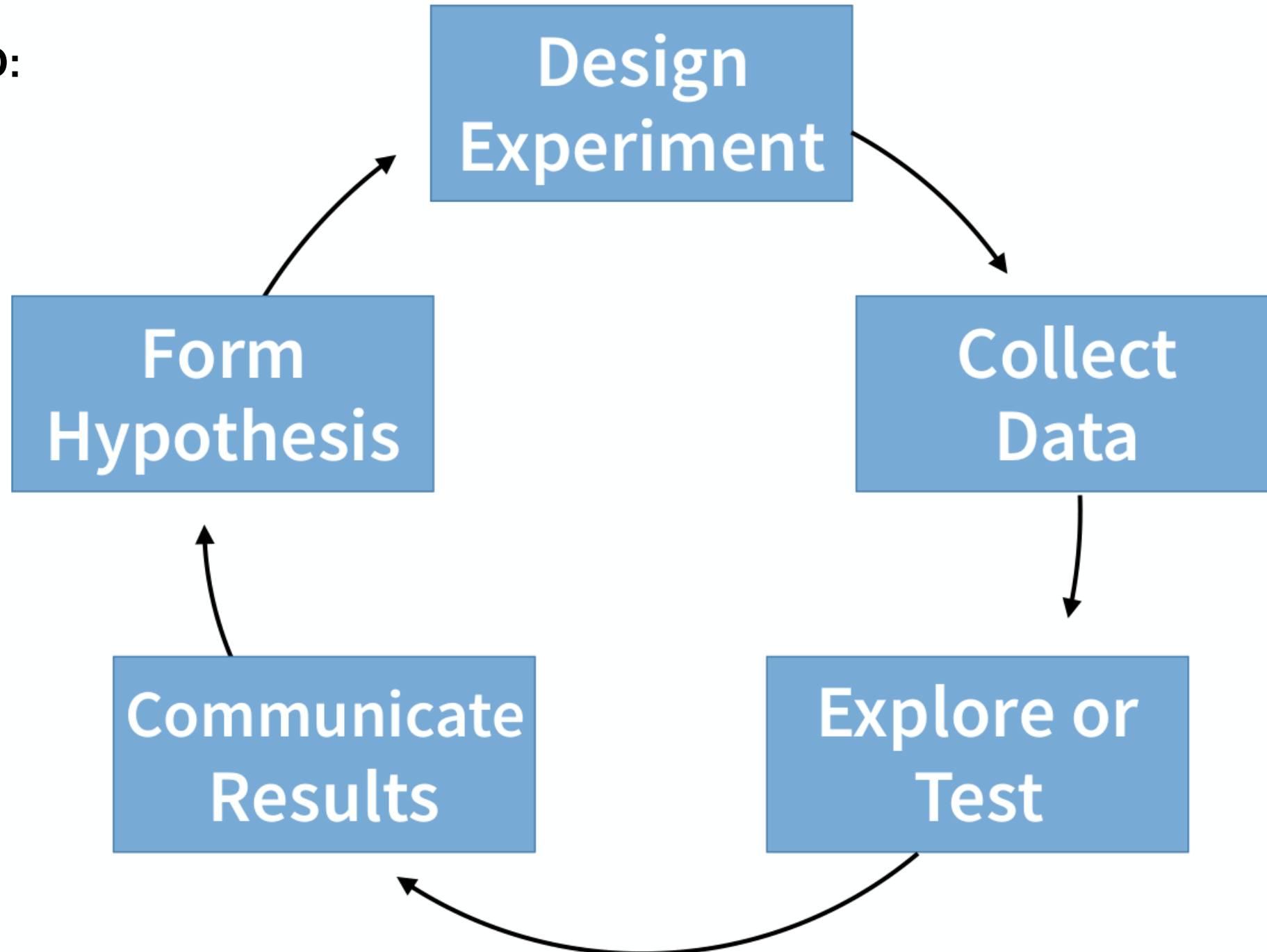


Tiempos

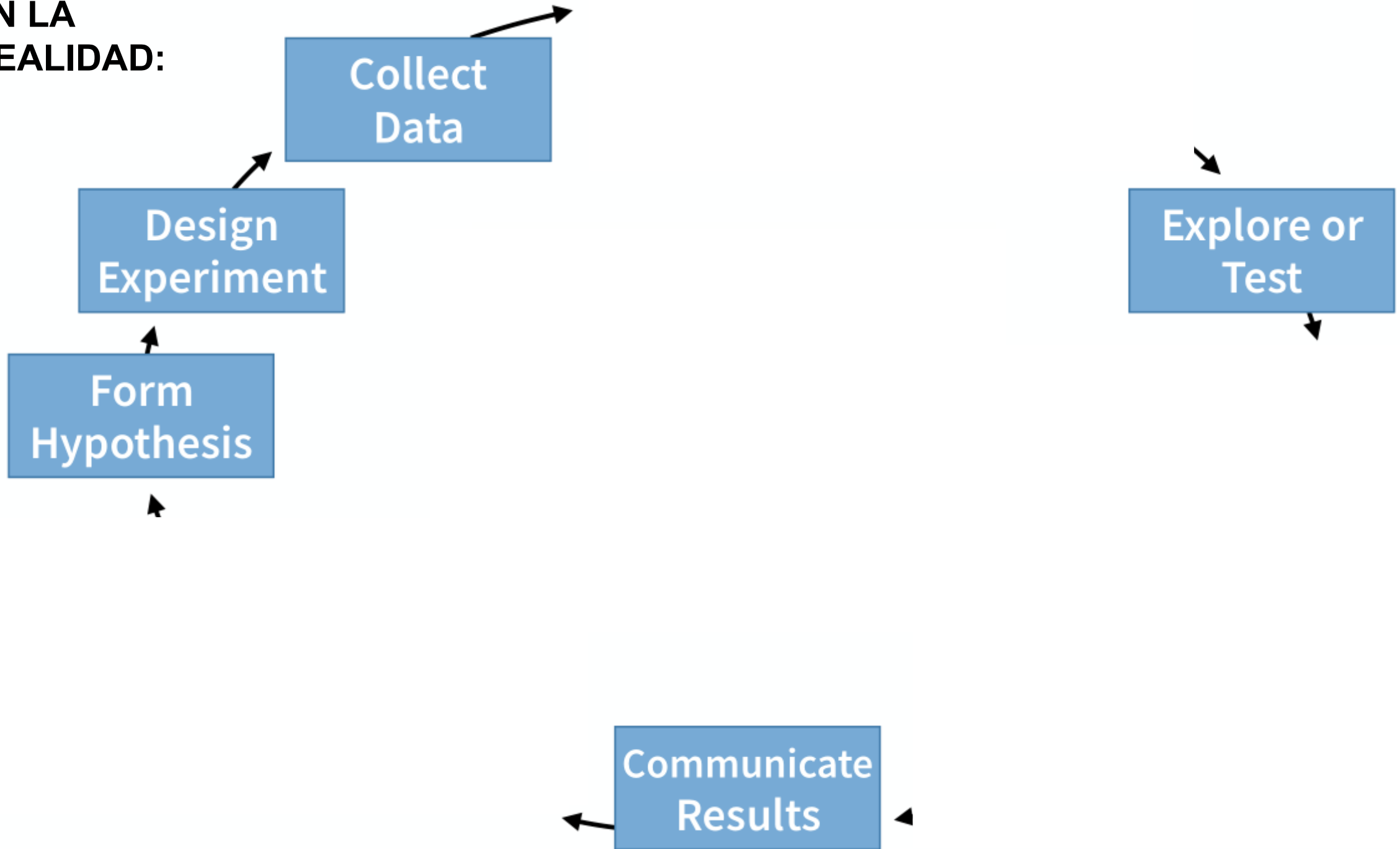
- **Introducción:** 25min
 - R vs Rstudio
 - conceptos clave
 - paquetes
- **Variables:** 30 min
 - concepto y clasificación
 - Tipos de datos en R
 - explorar distribuciones
- **Recapitulación:** 5min

R

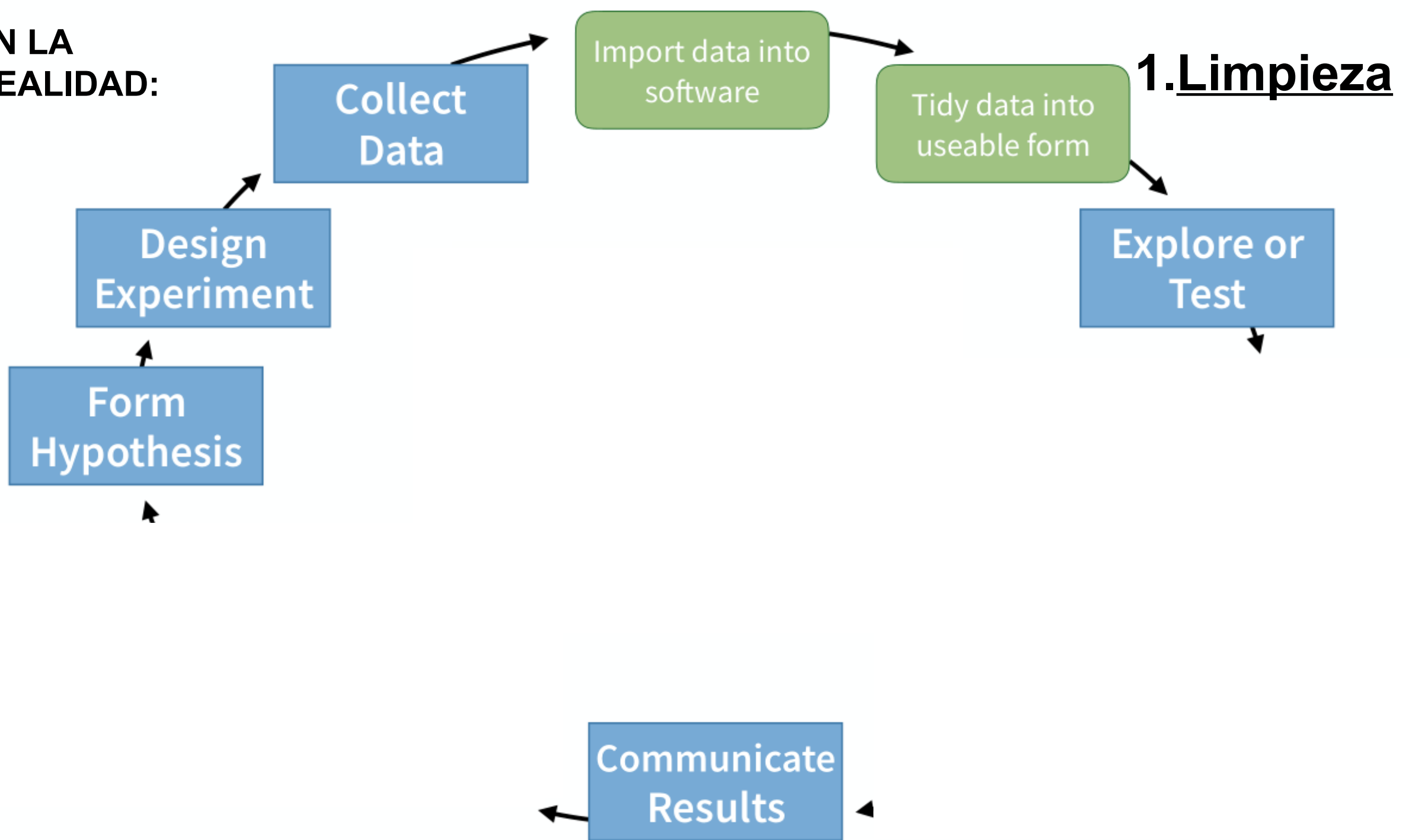
**MÉTODO
CIENTÍFICO:**



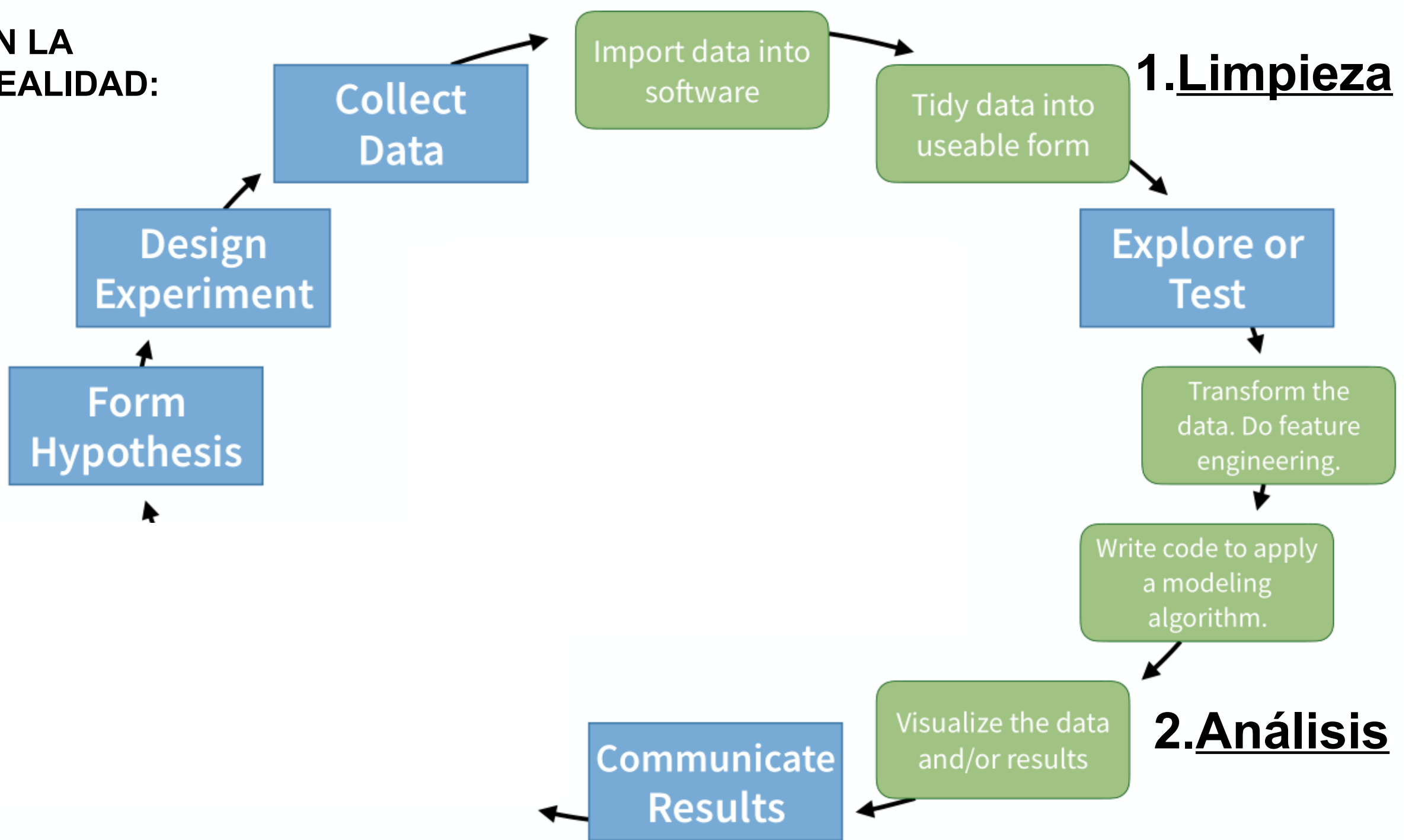
**EN LA
REALIDAD:**



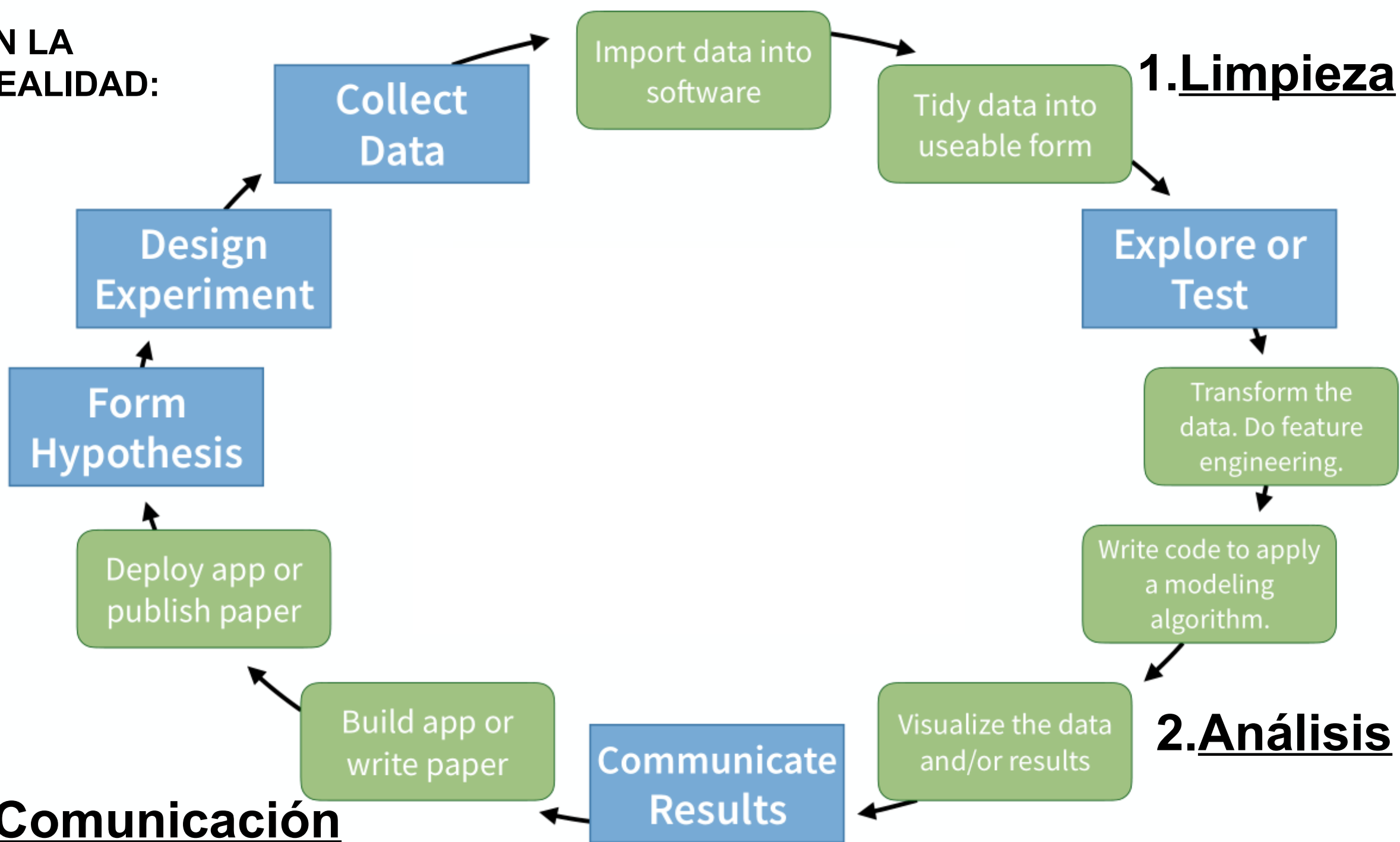
EN LA
REALIDAD:



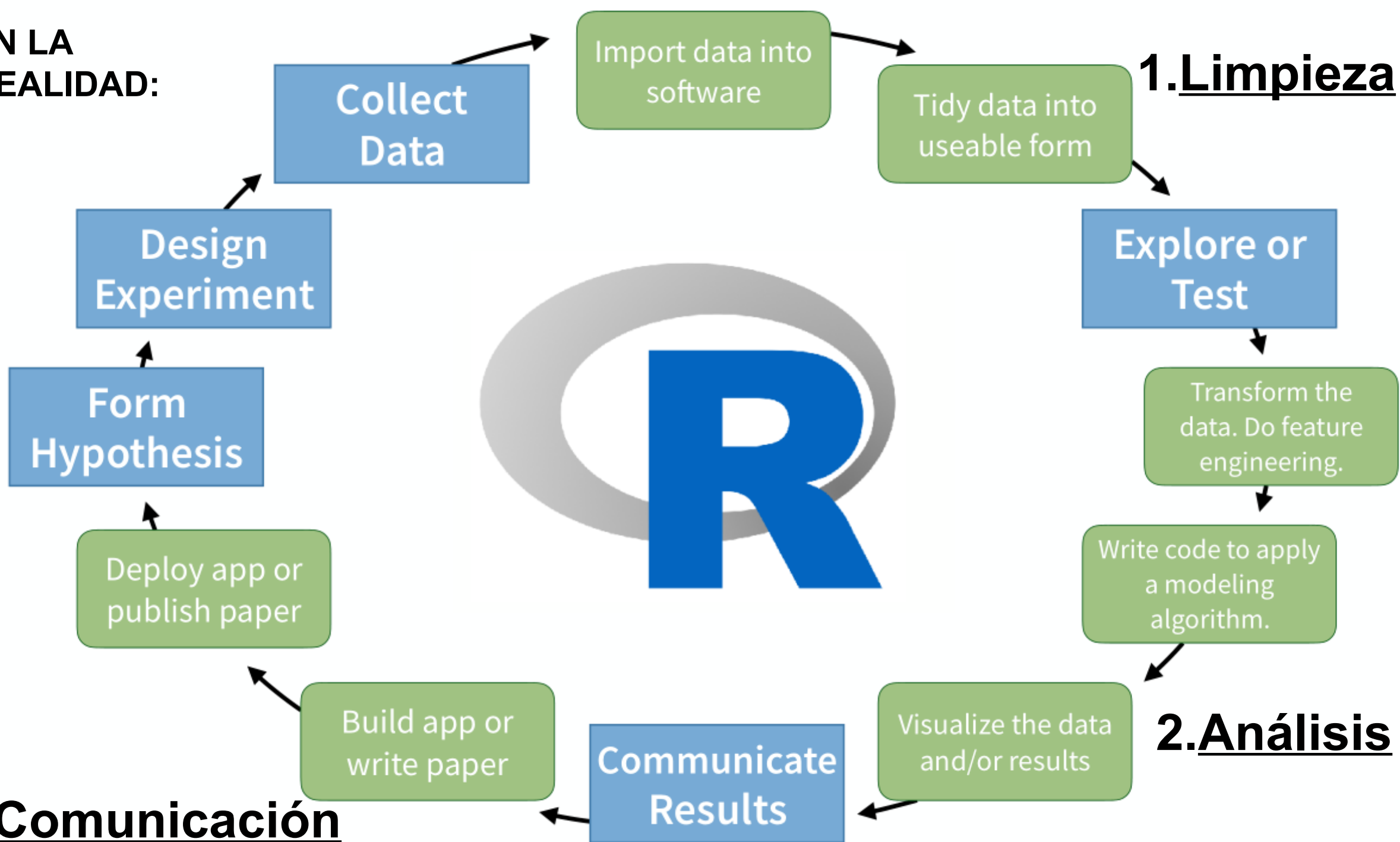
**EN LA
REALIDAD:**



**EN LA
REALIDAD:**



EN LA REALIDAD:



R: Do not open this



RStudio: Open this



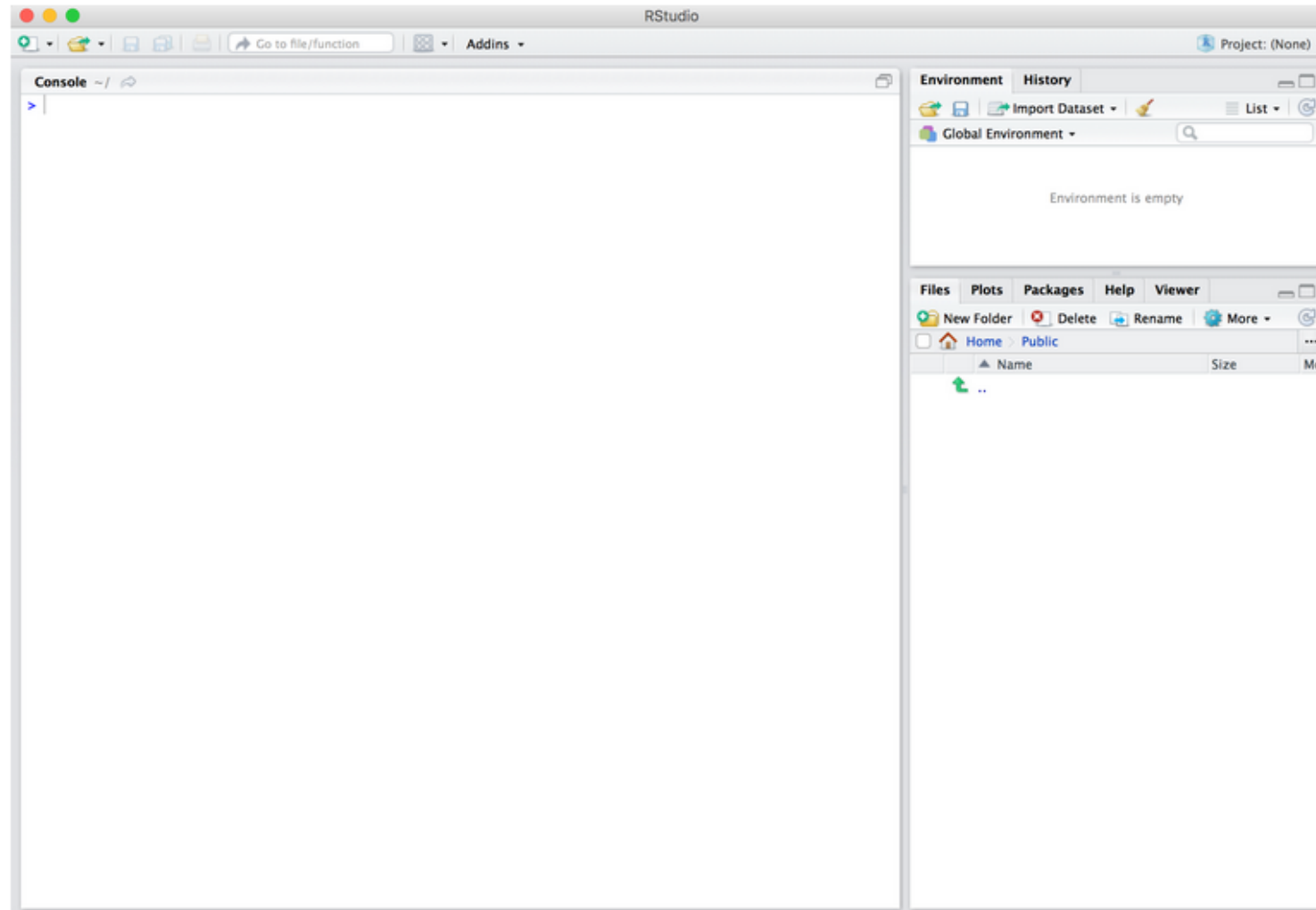
R: Do not open this



RStudio: Open this



After you open RStudio, you should see the following:



2.1 What are R and RStudio?

For much of this book, we will assume that you are using R via RStudio. First time users often confuse the two. At its simplest:

- R is like a car's engine.
- RStudio is like a car's dashboard.

R: Engine

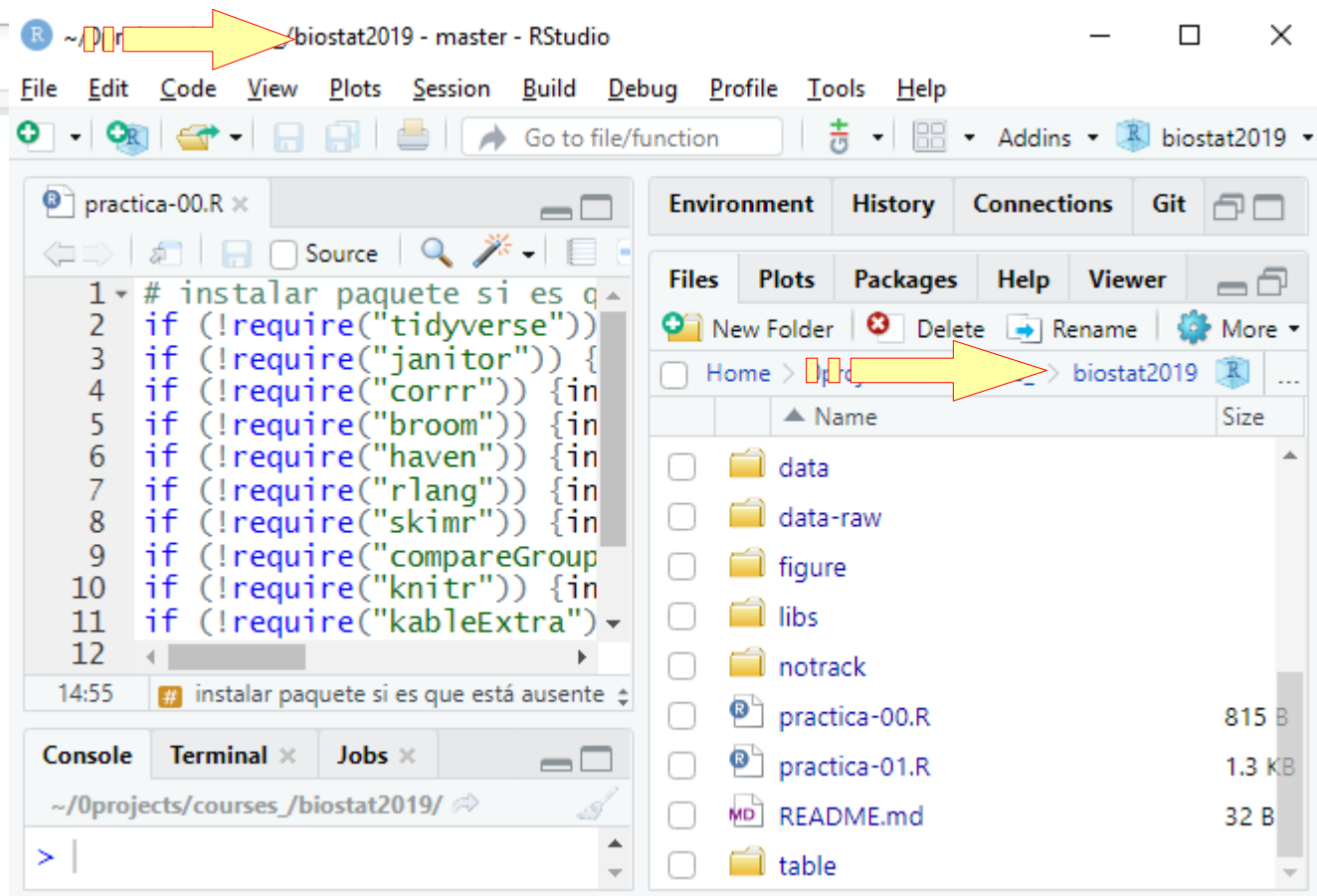
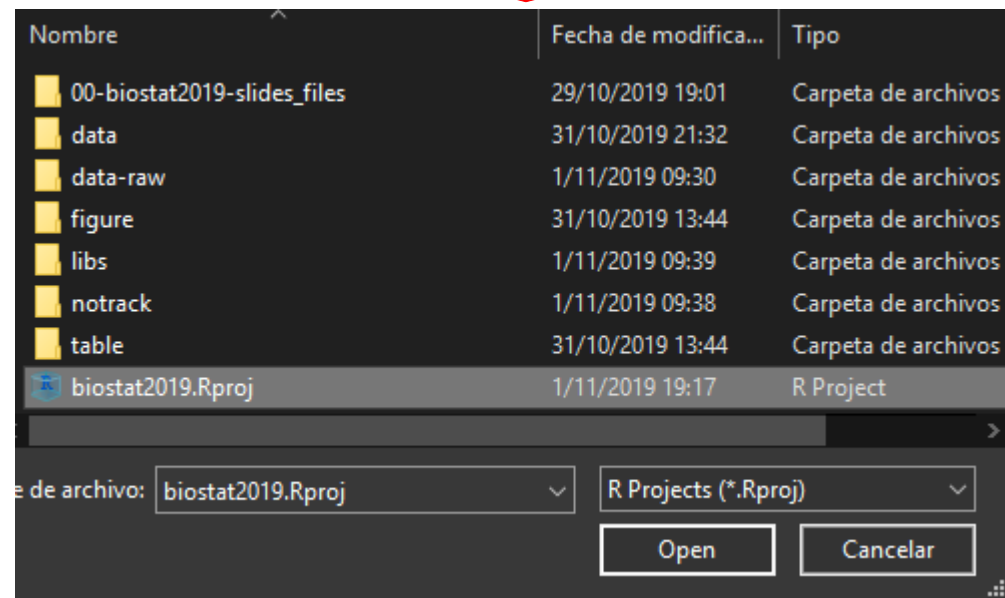
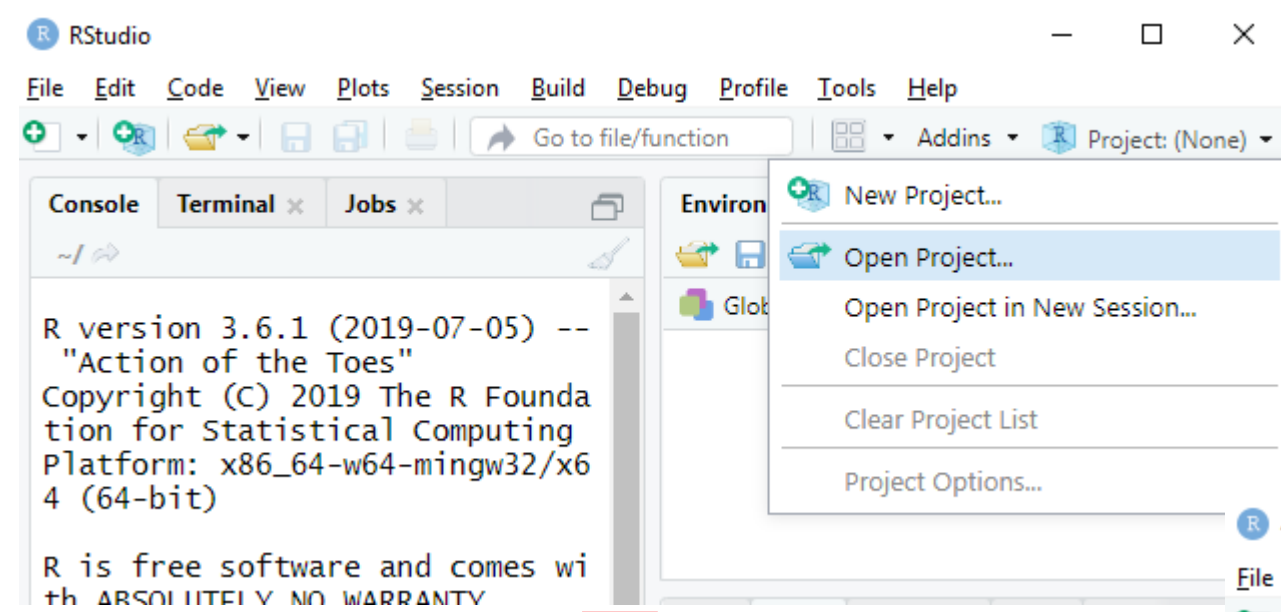


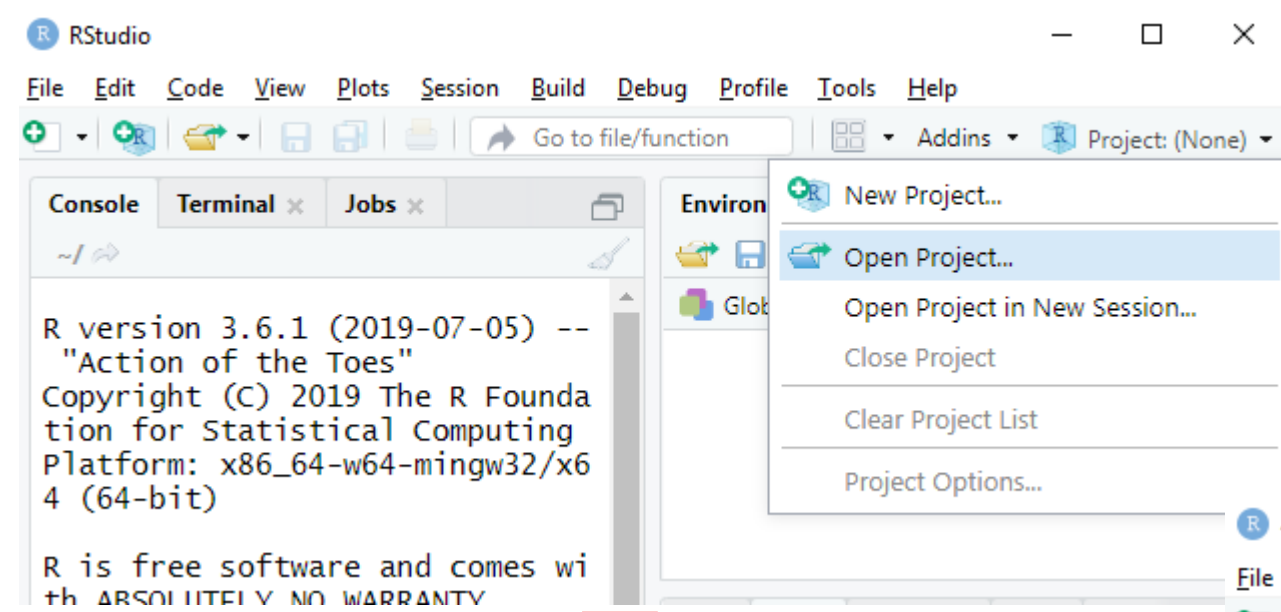
RStudio: Dashboard



¿Todos tienen el proyecto?

Organización *corriente abajo*





Ruta al proyecto es **única**
por computador

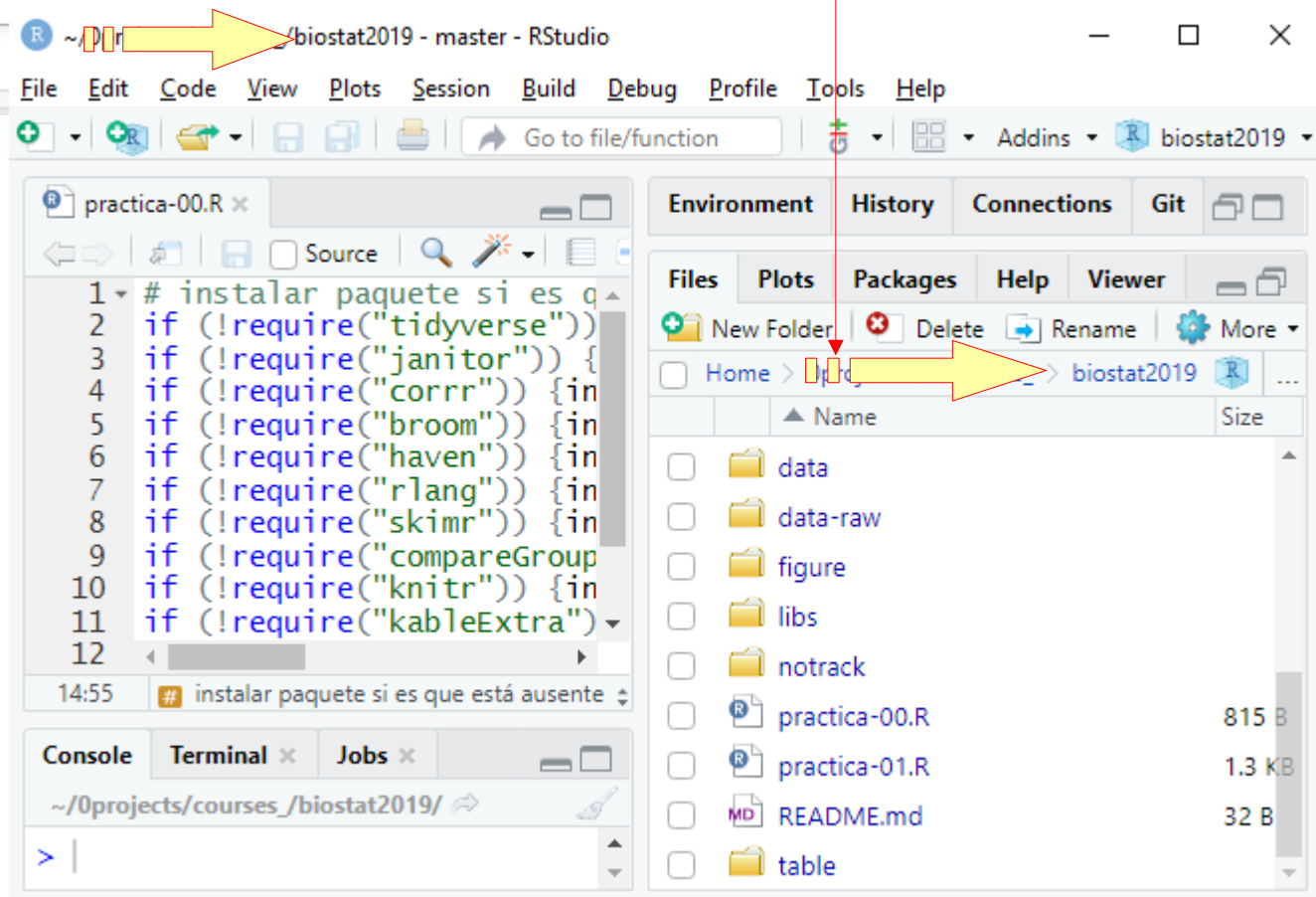
No afecta reproducibilidad

File explorer showing the contents of the biostat2019.Rproj project. The table lists files and folders with their names, modification dates, and types.

Nombre	Fecha de modifica...	Tipo
00-biostat2019-slides_files	29/10/2019 19:01	Carpeta de archivos
data	31/10/2019 21:32	Carpeta de archivos
data-raw	1/11/2019 09:30	Carpeta de archivos
figure	31/10/2019 13:44	Carpeta de archivos
libs	1/11/2019 09:39	Carpeta de archivos
notrack	1/11/2019 09:38	Carpeta de archivos
table	31/10/2019 13:44	Carpeta de archivos
biostat2019.Rproj	1/11/2019 19:17	R Project

File name: biostat2019.Rproj | File type: R Projects (*.Rproj)

Buttons: Open, Cancelar



Conceptos clave

Bases para interacción con R

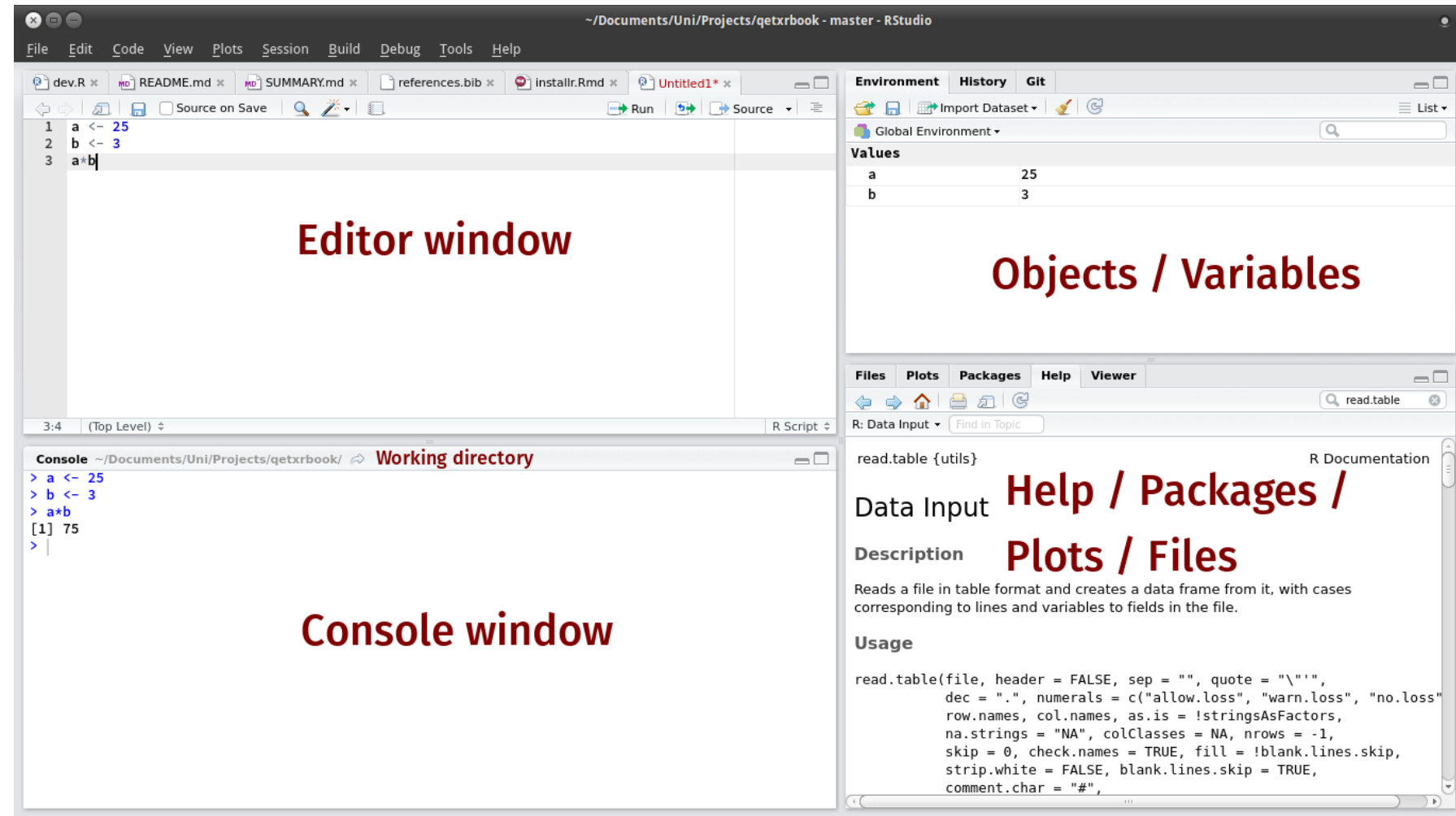
1. Editor y consola

---EDITOR---

```
a <- 5  
2+a
```

---CONSOLA---

```
> a <- 5  
> 2+a  
[1] 7
```



1. Editor y consola

---EDITOR---

```
a <- 5  
2+a
```

---CONSOLA---

```
> a <- 5  
> 2+a  
[1] 7
```

Comparar:

lo escrito en editor vs
lo impreso en consola

¿Cuál es la diferencia entre las dos
líneas?

2. Crear y ejecutar

---EDITOR----- ---COMENTARIOS---

a <- 5	Con <- <u>asignas</u> el número 5 al <u>objeto</u> a
2+a	Con + <u>ejecutas</u> la función suma con 2 y a

---CONSOLA--- ---COMENTARIOS---

> a <- 5	Esta acción <u>no</u> imprime resultado
> 2+a	Esta acción <u>sí</u> imprime resultado
[1] 7	Indica <u>orden</u> del 1er objeto en dicha línea

2. Crear y ejecutar

---EDITOR---

```
a <- 5
```

```
2+a
```

---CONSOLA---

```
> a <- 5
```

```
> 2+a
```

```
[1] 7
```

¡Hora de tomar nota!

¿Cómo escribir notas
en el editor

sin que R los lea
en la consola?

3. Comentarios y funciones

#comentario
función(objeto)

#todo contenido a la derecha de # no será ...
#ejecutado como función o “comando” en la consola de R

#p.e.: ¿cuál de las siguientes líneas será leída por R?

hist(mpg) #línea 1
#hist(mpg) #línea 2

3. Comentarios y funciones

---EDITOR---

a <- 5

2+a

3. Comentarios y funciones

---EDITOR---

---COMENTARIOS--- (Ctrl + Shift + R) (Alt + -)

```
a <- 5
```

```
#Con <- asignas el número 5 en el objeto a
```

```
#Esta acción no imprime resultado
```

```
2+a
```

```
#Con + ejecutas la función suma con 2 y a
```

```
#Esta acción sí imprime resultado
```

```
#El [1] indica orden del 1er objeto en dicha línea
```

4. Funciones y argumentos

```
función(argumento = "opción")  
función(argumento1 = "opción1", argumento2 = "opción2")
```

4. Funciones y argumentos

```
función(argumento = "opción")  
función(argumento1 = "opción1", argumento2 = "opción2")
```

#equivalente: si opción conserva orden de argumentos
función("opción1","opción2")

4. Funciones y argumentos

```
función(argumento = "opción")  
función(argumento1 = "opción1", argumento2 = "opción2")
```

#equivalente: si opción conserva orden de argumentos
función("opción1", "opción2")

#caso contrario: debes especificar el argumento
función(argumento2 = "opción2")

4. Funciones y argumentos

```
función(argumento = "opción")  
función(argumento1 = "opción1", argumento2 = "opción2")
```

#equivalente: si opción conserva orden de argumentos
función("opción1", "opción2")

#caso contrario: debes especificar el argumento
función(argumento2 = "opción2")


#puedes tener una función como opción de un argumento
función1(argumento1 = función2(argumento2 = "opción2"))

Práctica 1

Pipe

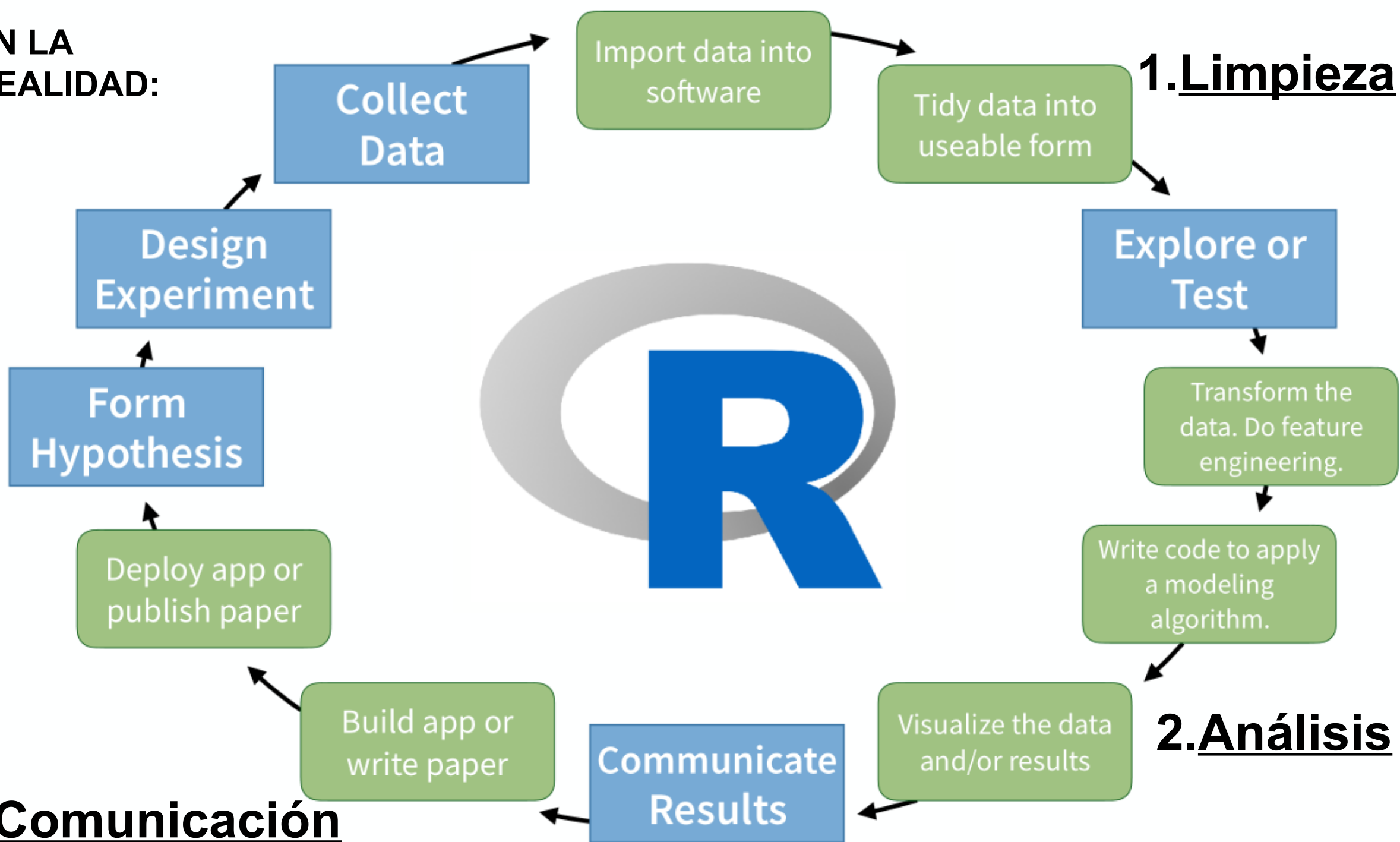
%>%

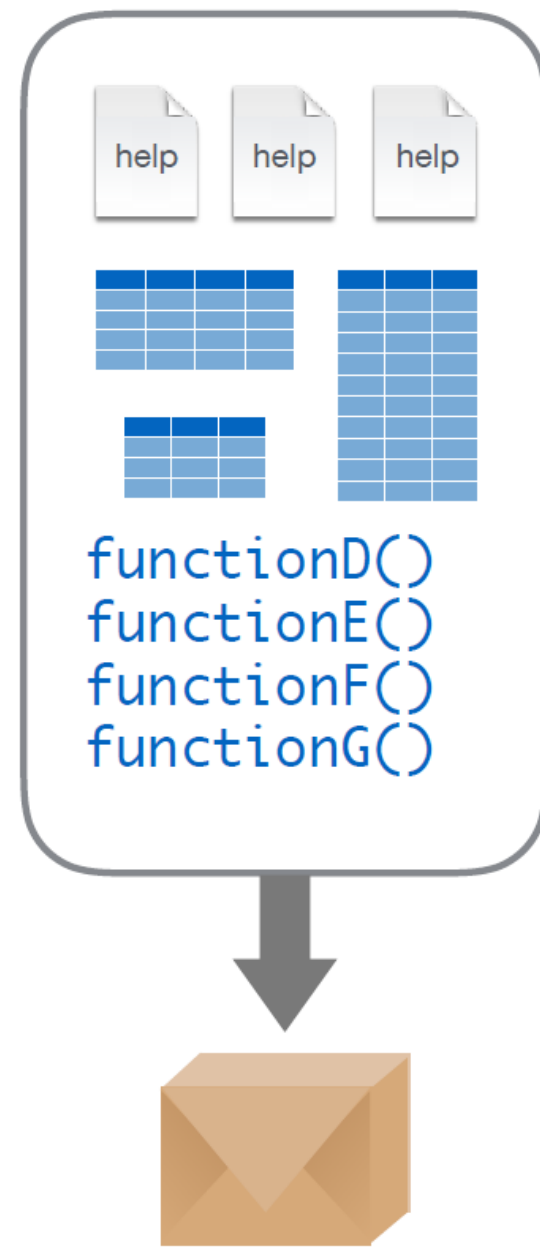
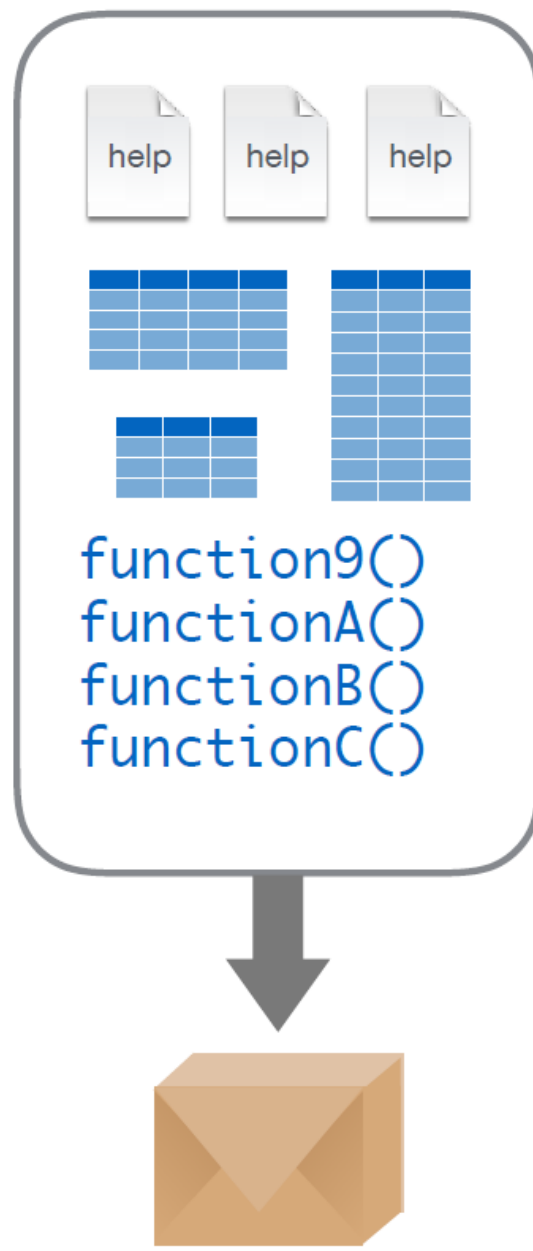
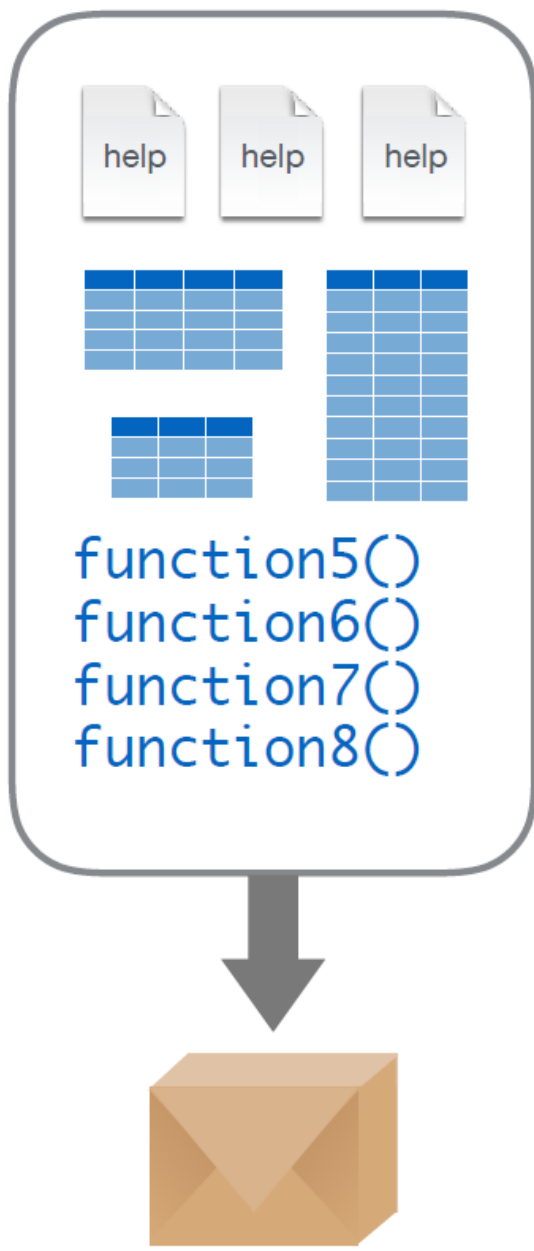
(Ctrl + Shift + M)



starwars %>% group_by(_____, species)

EN LA REALIDAD:





R Packages

Distribución libre de paquetes

A good analogy for R packages is they are like apps you can download onto a mobile phone:

R: A new phone



R Packages: Apps you can download



5. Paquetes: instalar e invocar

```
#instalar paquete (comillas obligatorias)  
install.package("paquete") #una vez por computador
```


5. Paquetes: instalar e invocar

#instalar paquete (comillas obligatorias)

install.package("paquete") #una vez por computador

#invocar paquete

library(paquete) #una vez por sesión

5. Paquetes: instalar e invocar

#instalar paquete (comillas obligatorias)

install.package("paquete") #una vez por computador

#invocar paquete

library(paquete) #una vez por sesión

#luego, estarás habilitado para usar/ejecutar función
función(argumento = "opción")

5. Paquetes: instalar e invocar

#instalar paquete (comillas obligatorias)

install.package("paquete") #una vez por computador

#invocar paquete

library(paquete) #una vez por sesión

#luego, estarás habilitado para usar/ejecutar función
función(argumento = "opción")

#si no invocaste el paquete con anticipación usar

paquete::función(argumento = "opción")

RECUERDA

1

```
install.packages("foo")
```

Downloads files to computer

1 x per computer

2

```
library("foo")
```

Loads package

1 x per R Session

6. Nombre de función repetido en \neq paquetes

6. Nombre de función repetido en ≠ paquetes

```
> library("tidyverse")
-- Attaching packages ----- tidyverse 1.2.1 --
v ggplot2 3.1.1          v purrr 0.3.0
v tibble 2.0.1           v dplyr 0.8.0.1
v tidyr 0.8.2            v stringr 1.4.0
v readr 1.3.1            v forcats 0.4.0
-- Conflicts ----- tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag() masks stats::lag()
```

6. Nombre de función repetido en ≠ paquetes

```
> library("tidyverse")
-- Attaching packages ----- tidyverse 1.2.1 --
v ggplot2 3.1.1      v purrr 0.3.0
v tibble 2.0.1       v dplyr 0.8.0.1
v tidyr 0.8.2        v stringr 1.4.0
v readr 1.3.1        v forcats 0.4.0
-- Conflicts ----- tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag() masks stats::lag()
```

6. Nombre de función repetido en ≠ paquetes

```
> library("tidyverse")
```

```
-- Attaching packages ----- tidyverse 1.2.1 --  
v ggplot2 3.1.1      v purrr 0.3.0  
v tibble 2.0.1       v dplyr 0.8.0.1  
v tidyr 0.8.2        v stringr 1.4.0  
v readr 1.3.1        v forcats 0.4.0  
  
-- Conflicts ----- tidyverse_conflicts() --  
x dplyr::filter() masks stats::filter()  
x dplyr::lag()    masks stats::lag()
```

#ejemplo:

filter(argumento1 = "opción1") #¿qué paquete usará?

6. Nombre de función repetido en ≠ paquetes

```
> library("tidyverse")
```

```
-- Attaching packages ----- tidyverse 1.2.1 --  
v ggplot2 3.1.1      v purrr 0.3.0  
v tibble 2.0.1       v dplyr 0.8.0.1  
v tidyr 0.8.2        v stringr 1.4.0  
v readr 1.3.1        v forcats 0.4.0  
  
-- Conflicts ----- tidyverse_conflicts() --  
x dplyr::filter() masks stats::filter()  
x dplyr::lag()    masks stats::lag()
```

#ejemplo:

```
filter(argumento1 = "opción1") #usará paquete dplyr
```

6. Nombre de función repetido en ≠ paquetes

```
> library("tidyverse")
```

```
-- Attaching packages ----- tidyverse 1.2.1 --  
v ggplot2 3.1.1      v purrr 0.3.0  
v tibble 2.0.1       v dplyr 0.8.0.1  
v tidyr 0.8.2        v stringr 1.4.0  
v readr 1.3.1        v forcats 0.4.0  
  
-- Conflicts ----- tidyverse_conflicts() --  
x dplyr::filter() masks stats::filter()  
x dplyr::lag()    masks stats::lag()
```

#ejemplo:

filter(argumento1 = "opción1") #usará paquete dplyr

stat::filter(argumento2 = "opción2") #usará paquete **stat**

Variables

Concepto y clasificación

Table 3.1 List of important terms. Examples pertain to a hypothetical research investigation into estimating the protein content of koala milk.

Term	Definition	Example
<i>Measurement</i>	A single piece of recorded information reflecting a characteristic of interest (e.g. length of a leaf, pH of a water aliquot mass of an individual, number of individuals per quadrat etc)	Protein content of the milk of a single female koala
<i>Observation</i>	A single measured sampling or experimental unit (such as an individual, a quadrat, a site etc)	A small quantity of milk from a single koala
<i>Population</i>	All the possible observations that could be measured and the unit of which wish to draw conclusions about (note a statistical population need not be a viable biological population)	The milk of all female koalas
<i>Sample</i>	The (representative) subset of the population that are observed	A small quantity of milk collected from 15 captive female koalas ^a
<i>Variable</i>	A set of measurements of the same type that comprise the sample. The characteristic that differs (varies) from observation to observation	The protein content of koala milk.

^a Note that such a sample may not actually reflect the defined population. Rather, it could be argued that such a sample reflects captive populations. Nevertheless, such extrapolations are common when field samples are difficult to obtain.

1. Variables: según dependencia

1. Dependientes (desenlace)

- Variables a explicar (p.e. cáncer al pulmón) respecto a los cuales hay que buscar un motivo.

2. Independientes (exposición)

- Variable que podría explicar un cambio (p.e. fumar) en los valores de la variable dependiente.

3. Intervenientes

- Afectan la relación: var.dependiente - independiente
- Confusor (edad) o Modificador de efecto (ejercicio)

2. Variables: según naturaleza

1. Numérica (cuantitativa)

- Continua:

p.e. altura (m) 1.35 ; 1.46 ; 2.05 ; ...

- Discreta:

p.e. número de RNA-seq reads \rightarrow 0 ; 448 ; 633; ...

2. Categórica (cualitativa)

- Dicotómica:

p.e. paciente sintomático o asintomático

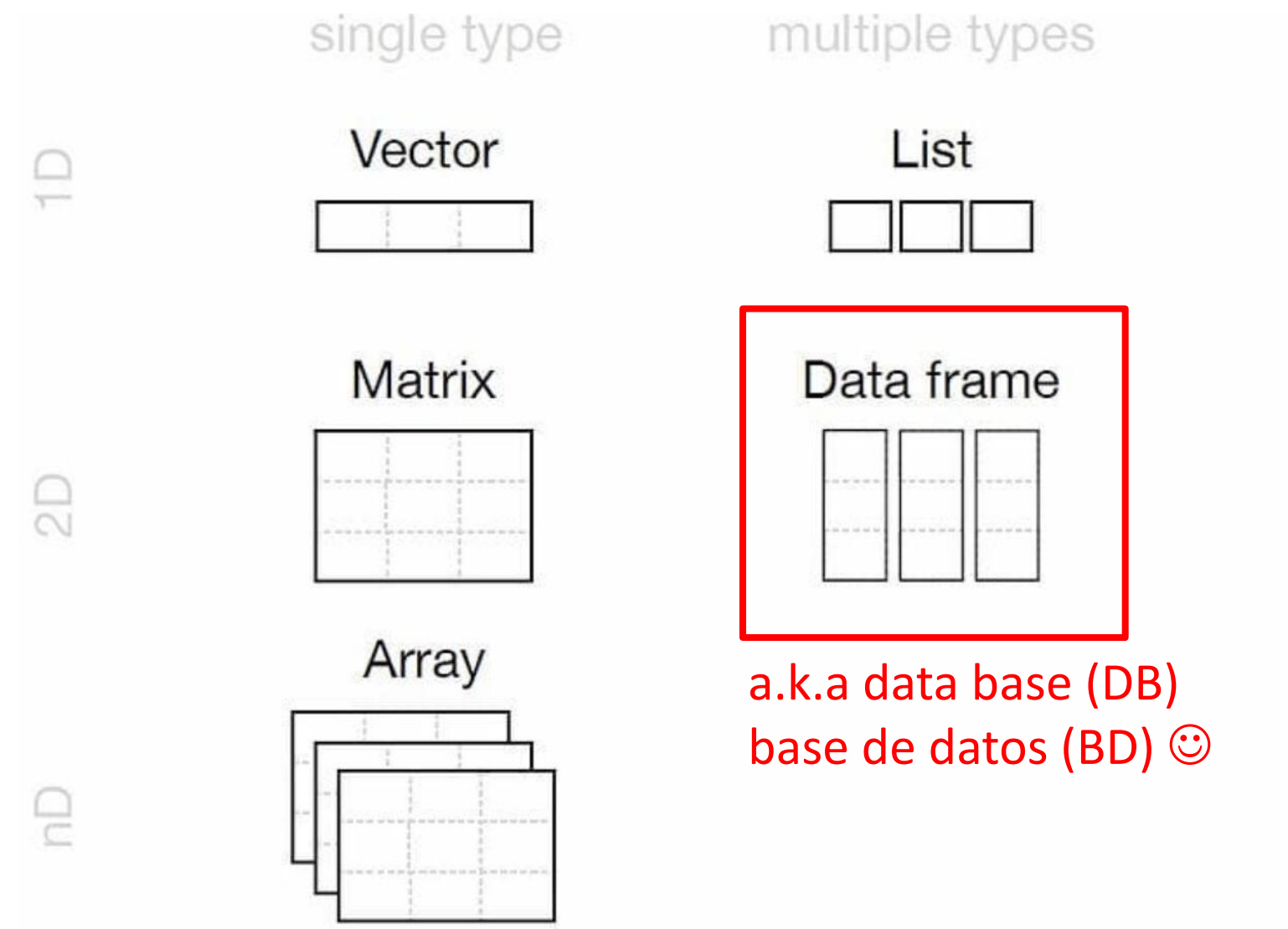
- Politómica:

p.e. grupo sanguíneo A, B, AB, 0

R data types

De vectores a bases de datos

6. Tipos de datos en R



7. Tipos de vectores en un Data Frame o DB

```
> library(gapminder)
> gapminder
# A tibble: 1,704 x 6
  country      continent  year lifeExp      pop gdpPercap
  <fct>        <fct>    <int>  <dbl>    <int>    <dbl>
1 Afghanistan Asia      1952   28.8  8425333    779.
2 Afghanistan Asia      1957   30.3  9240934    821.
3 Afghanistan Asia      1962   32.0 10267083    853.
4 Afghanistan Asia      1967   34.0 11537966    836.
5 Afghanistan Asia      1972   36.1 13079460    740.
6 Afghanistan Asia      1977   38.4 14880372    786.
7 Afghanistan Asia      1982   39.9 12881816    978.
8 Afghanistan Asia      1987   40.8 13867957    852.
9 Afghanistan Asia      1992   41.7 16317921    649.
10 Afghanistan Asia      1997   41.8 22227415    635.
```


7. Tipos de vectores en un Data Frame o DB

```
> library(gapminder)
```

```
> gapminder
```

```
# A tibble: 1,704 x 6
```

Dimensiones: N°filas x N°columnas



	country	continent	year	lifeExp	pop	gdpPercap
	<fct>	<fct>	<int>	<dbl>	<int>	<dbl>
1	Afghanistan	Asia	<u>1952</u>	28.8	8 <u>425333</u>	779.
2	Afghanistan	Asia	<u>1957</u>	30.3	9 <u>240934</u>	821.
3	Afghanistan	Asia	<u>1962</u>	32.0	10 <u>267083</u>	853.
4	Afghanistan	Asia	<u>1967</u>	34.0	11 <u>537966</u>	836.
5	Afghanistan	Asia	<u>1972</u>	36.1	13 <u>079460</u>	740.
6	Afghanistan	Asia	<u>1977</u>	38.4	14 <u>880372</u>	786.
7	Afghanistan	Asia	<u>1982</u>	39.9	12 <u>881816</u>	978.
8	Afghanistan	Asia	<u>1987</u>	40.8	13 <u>867957</u>	852.
9	Afghanistan	Asia	<u>1992</u>	41.7	16 <u>317921</u>	649.
10	Afghanistan	Asia	<u>1997</u>	41.8	22 <u>227415</u>	635.

7. Tipos de vectores en un Data Frame o DB

```
> library(gapminder)
```

```
> gapminder
```

```
# A tibble: 1,704 x 6
```

	country	continent	year	lifeExp	pop	gdpPercap
	<fct>	<fct>	<int>	<dbl>	<int>	<dbl>
1	Afghanistan	Asia	1952	28.8	8425333	779.
2	Afghanistan	Asia	1957	30.3	9240934	821.
3	Afghanistan	Asia	1962	32.0	10267083	853.
4	Afghanistan	Asia	1967	34.0	11537966	836.
5	Afghanistan	Asia	1972	36.1	13079460	740.
6	Afghanistan	Asia	1977	38.4	14880372	786.
7	Afghanistan	Asia	1982	39.9	12881816	978.
8	Afghanistan	Asia	1987	40.8	13867957	852.
9	Afghanistan	Asia	1992	41.7	16317921	649.
10	Afghanistan	Asia	1997	41.8	22227415	635.

Dimensiones: N°filas x N°columnas



- Una DB contiene múltiples vectores

- Cada columna es un vector

- Cada vector es una variable

- Cada tipo de vector corresponde a un tipo de variable

7. Tipos de vectores en un Data Frame o DB

```
> library(gapminder)
```

```
> gapminder
```

```
# A tibble: 1,704 x 6
```

	country	continent	year	lifeExp	pop	gdpPercap
	<fct>	<fct>	<int>	<dbl>	<int>	<dbl>
1	Afghanistan	Asia	1952	28.8	8425333	779.
2	Afghanistan	Asia	1957	30.3	9240934	821.
3	Afghanistan	Asia	1962	32.0	10267083	853.
4	Afghanistan	Asia	1967	34.0	11537966	836.
5	Afghanistan	Asia	1972	36.1	13079460	740.
6	Afghanistan	Asia	1977	38.4	14880372	786.
7	Afghanistan	Asia	1982	39.9	12881816	978.
8	Afghanistan	Asia	1987	40.8	13867957	852.
9	Afghanistan	Asia	1992	41.7	16317921	649.
10	Afghanistan	Asia	1997	41.8	22227415	635.

Dimensiones: N°filas x N°columnas



factor

integer

double

(categórica)

(numérica discreta)

(numérica continua)

- Una DB contiene múltiples vectores

- Cada columna es un vector

- Cada vector es una variable

- Cada tipo de vector corresponde a un tipo de variable

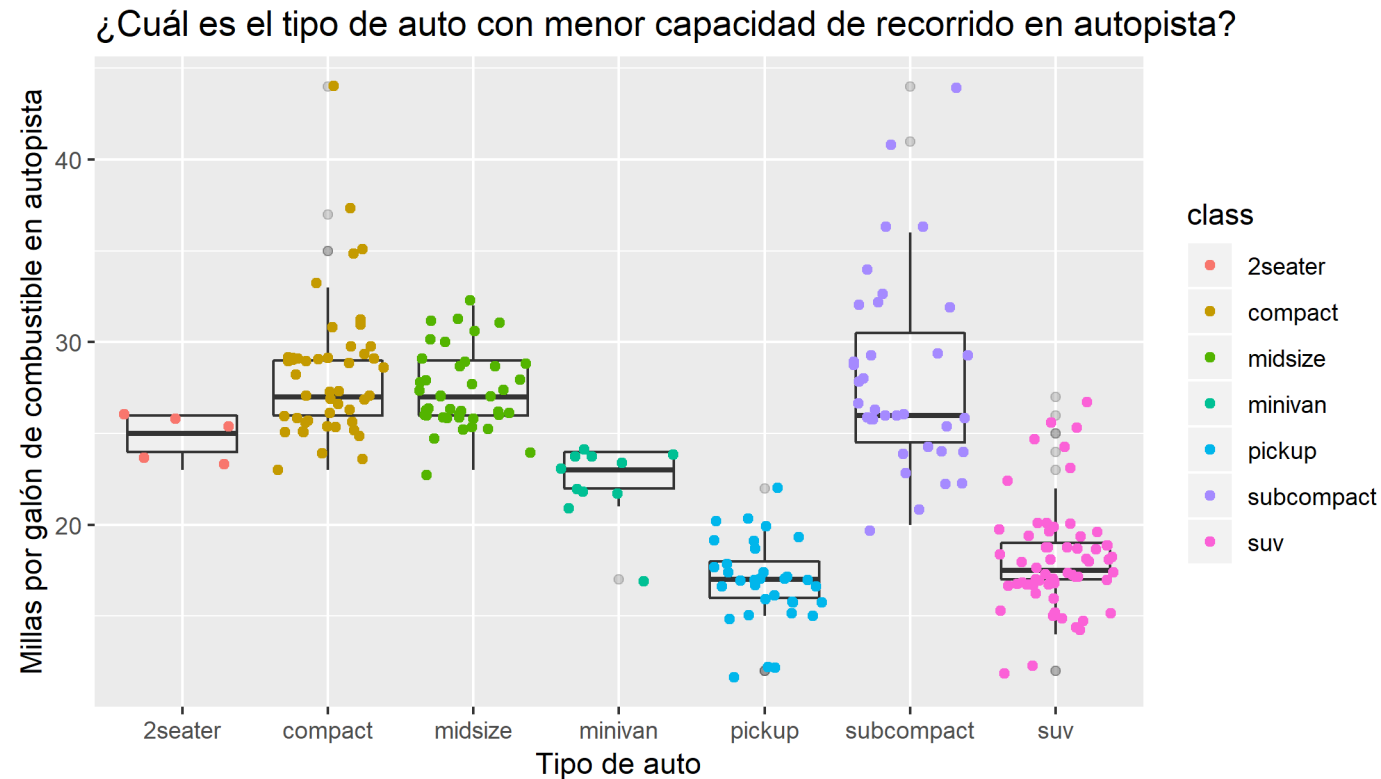
Explorar distribuciones

retornar a
parte 01

Visualización de datos

ejemplo

```
ggplot(mpg, aes(class, hwy)) +  
  geom_boxplot(alpha=0.2) +  
  geom_point(aes(color=class), position = "jitter")
```



Fuente: base de datos mpg
(mpg = millas por galón)