

# Sesión 1: Introducción a RMarkdown y Shiny Web App

## Aplicaciones en Computación Estadística

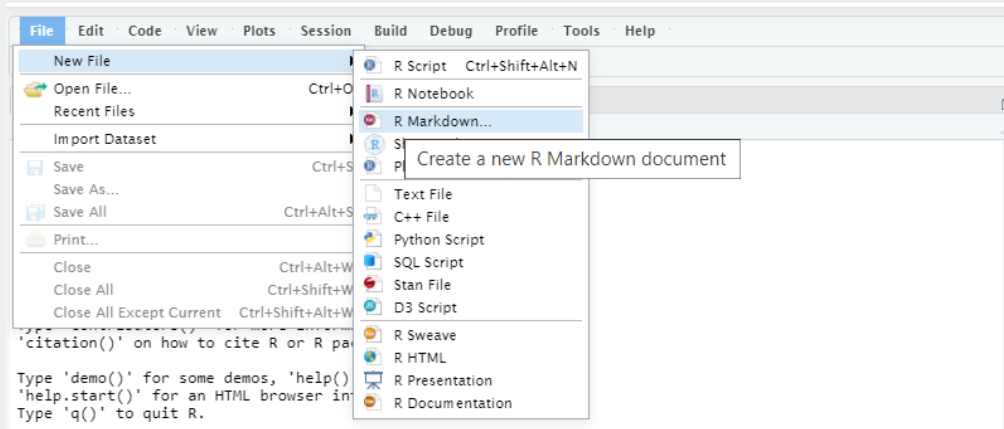
Natalie Julian - [www.nataliejulian.com](http://www.nataliejulian.com)

Estadística UC y Data Scientist en Zippedi Inc.

# RMarkdown

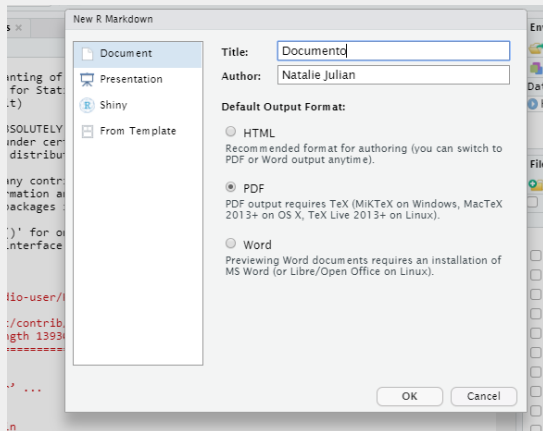
Generar documentos en R es de muchísima utilidad para poder reportar nuestros resultados de una manera más eficiente y limpia.

# Creando un archivo RMarkdown



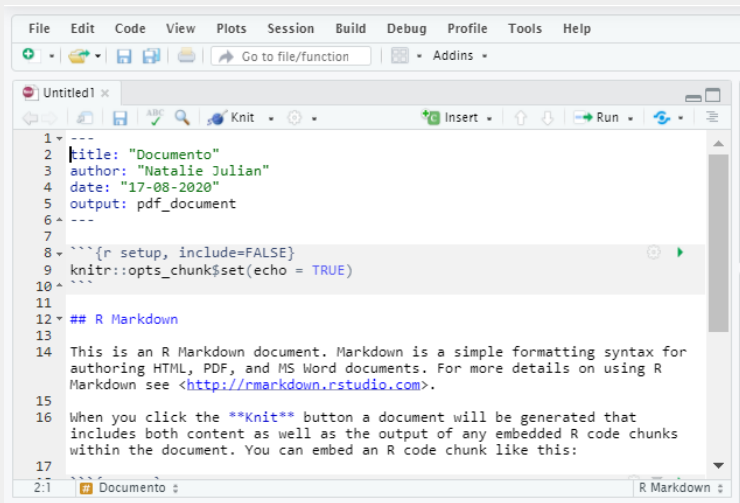
# Creando un archivo RMarkdown

Podemos añadir título y nombre del archivo y seleccionar el formato de salida del informe. Utilizaremos PDF para crear un informe plano:



# Código de RMarkdown

Al crear el archivo se abre el siguiente código base para un informe en RMarkdown:



The screenshot shows the RStudio application window with a new R Markdown document titled 'Untitled1'. The menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. The toolbar contains icons for creating a new file, opening a file, saving, and navigating. The main editor area displays the following R Markdown code template:

```
1 ---
2 title: "Documento"
3 author: "Natalie Julian"
4 date: "17-08-2020"
5 output: pdf_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for
15 authoring HTML, PDF, and MS Word documents. For more details on using R
16 Markdown see <http://rmarkdown.rstudio.com>.
17
18 When you click the **Knit** button a document will be generated that
19 includes both content as well as the output of any embedded R code chunks
20 within the document. You can embed an R code chunk like this:
```

The status bar at the bottom indicates the current line is 2:1 and the document type is R Markdown.

Para compilar el archivo, hacemos click en Knit.

## Documento

Natalie Julian

17-08-2020

### R Markdown

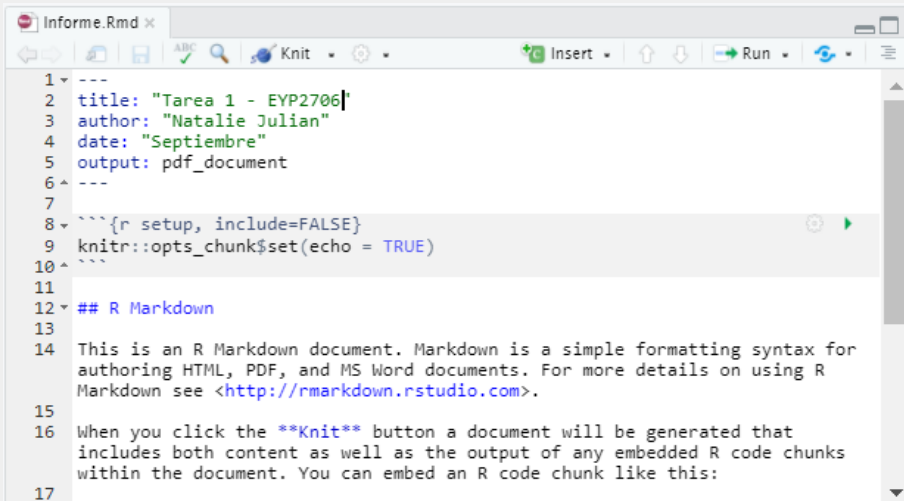
This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

##	speed	dist
##	Min. : 4.0	Min. : 2.00
##	1st Qu.:12.0	1st Qu.: 26.00
##	Median :15.0	Median : 36.00
##	Mean :15.4	Mean : 42.98
##	3rd Qu.:19.0	3rd Qu.: 56.00
##	Max. :25.0	Max. :120.00

# Modificando detalles



```
1 ---
2 title: "Tarea 1 - EYP2706"
3 author: "Natalie Julian"
4 date: "Septiembre"
5 output: pdf_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for
15 authoring HTML, PDF, and MS Word documents. For more details on using R
16 Markdown see <http://rmarkdown.rstudio.com>.
17
18 When you click the Knit button a document will be generated that
19 includes both content as well as the output of any embedded R code chunks
20 within the document. You can embed an R code chunk like this:
```



## Tarea 1 - EYP2706

Natalie Julian

Septiembre

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed      dist
##  Min.   : 4.0    Min.   : 2.00
##  1st Qu.:12.0    1st Qu.: 26.00
##  Median :15.0    Median : 36.00
##  Mean   :15.4    Mean   : 42.98
##  3rd Qu.:19.0    3rd Qu.: 56.00
##  Max.   :25.0    Max.   :120.00
```

a) Añada una sección en el informe llamada *Actividad 1*.

Para crear secciones, subsecciones y subsubsecciones, se utiliza la siguiente estructura:

# Sección

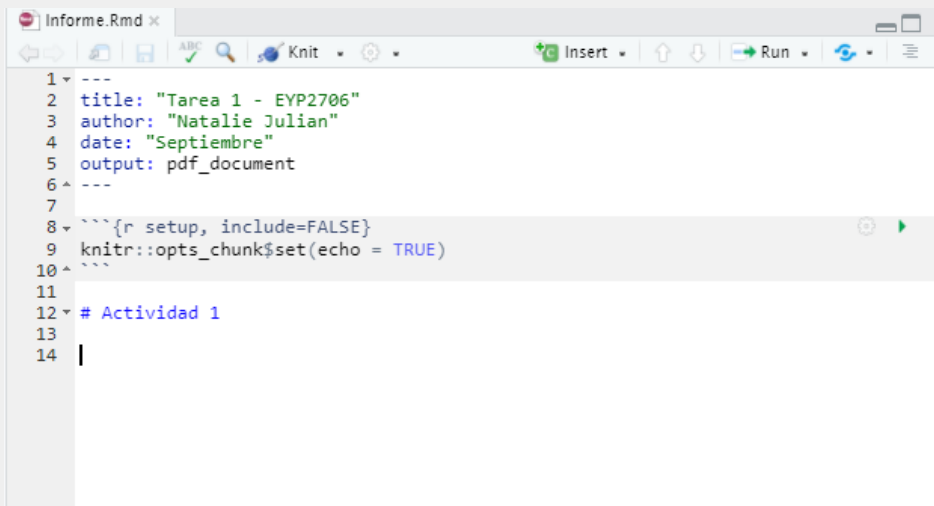
## Subsección

### Subsubsección

#### Subsubsubsección

Y así sucesivamente.

# Sección Actividad 1



```
1 ---
2 title: "Tarea 1 - EYP2706"
3 author: "Natalie Julian"
4 date: "Septiembre"
5 output: pdf_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 # Actividad 1
13
14 |
```

## Tarea 1 - EYP2706

Natalie Julian

Septiembre

### Actividad 1

Crearemos un informe utilizando la base de datos `hotel`.

b) Añada la siguiente descripción del problema:

La base de datos *hotel* contiene información sobre reservaciones de distintos hoteles. Interesa estudiar la variable target *Cancel* la que indica si la reservación fue o no cancelada.

- ▶ **Cancel:** Indica si la reserva fue cancelada o no (1: Cancelada, 0: No cancelada)
- ▶ **Mes:** Mes de la reserva
- ▶ **Weekend:** Cantidad de días de fin de semana (Sábados o Domingos) reservados
- ▶ **Weekday:** Cantidad de días de la semana (Lunes, Martes, Miércoles, Jueves, Viernes) reservados

```
6 ^ ---
7
8 v ```{r setup, include=FALSE}
9   knitr::opts_chunk$set(echo = TRUE)
10 ^ ```
11
12 v # Actividad 1
13
14   La base de datos _hotel_ contiene información sobre reservaciones de
15   distintos hoteles. Interesa estudiar la variable target _Cancel_ la que
16   indica si la reservación fue o no cancelada.
17
18   - **Cancel**: Indica si la reserva fue cancelada o no (1: Cancelada, 0: No
19     cancelada)
20   - **Mes**: Mes de la reserva
21   - **Weekend**: Cantidad de días de fin de semana (Sábados o Domingos)
22     reservados
23   - **Weekday**: Cantidad de días de la semana (Lunes, Martes, Miércoles,
24     Jueves, Viernes) reservados
```

18:81 # Actividad 1 ↕

R Markdown ↕

## Tarea 1 - EYP2706

Natalie Julian

Septiembre

### Actividad 1

La base de datos *hotel* contiene información sobre reservaciones de distintos hoteles. Interesa estudiar la variable target *Cancel* la que indica si la reservación fue o no cancelada.

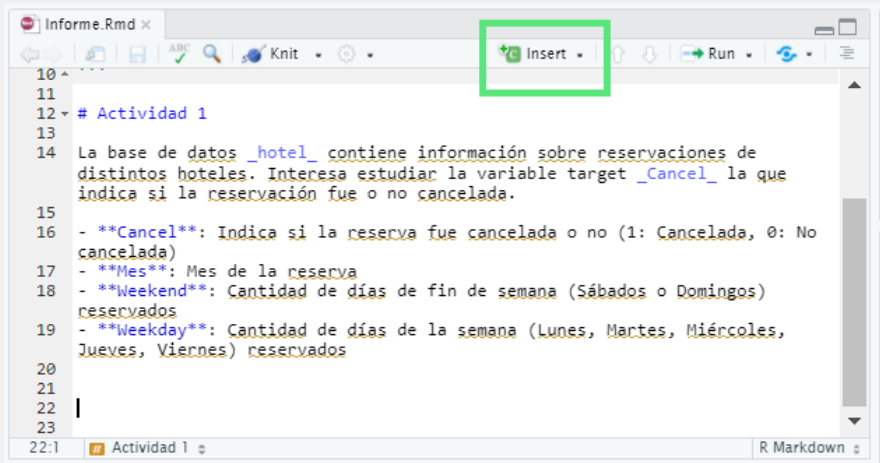
- **Cancel:** Indica si la reserva fue cancelada o no (1: Cancelada, 0: No cancelada)
- **Mes:** Mes de la reserva
- **Weekend:** Cantidad de días de fin de semana (Sábados o Domingos) reservados
- **Weekday:** Cantidad de días de la semana (Lunes, Martes, Miércoles, Jueves, Viernes) reservados

c) Cargue la base de datos. Y realice análisis descriptivo de ésta.



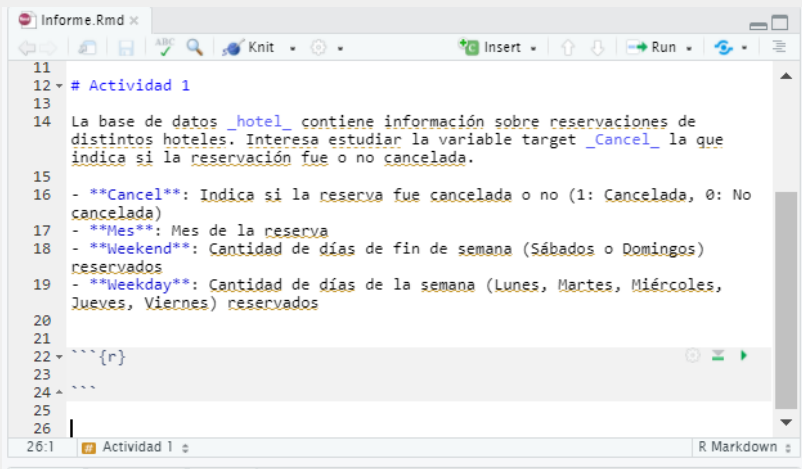
Para cargar la base de datos, necesitamos abrir un chunk (o pedazo de código) e incluir el código de importación en este chunk.

# Creando un chunk



```
Informe.Rmd x
10
11
12 # Actividad 1
13
14 La base de datos _hotel_ contiene información sobre reservaciones de
distintos hoteles. Interesa estudiar la variable target _Cancel_ la que
indica si la reservación fue o no cancelada.
15
16 - **Cancel**: Indica si la reserva fue cancelada o no (1: Cancelada, 0: No
cancelada)
17 - **Mes**: Mes de la reserva
18 - **Weekend**: Cantidad de días de fin de semana (Sábados o Domingos)
reservados
19 - **Weekday**: Cantidad de días de la semana (Lunes, Martes, Miércoles,
Jueves, Viernes) reservados
20
21
22 |
23
22:1 Actividad 1 R Markdown
```

# Vista del chunk



The screenshot shows an R Markdown editor window titled 'Informe.Rmd'. The editor displays a code chunk for 'Actividad 1' starting at line 11. The chunk contains a title, a paragraph, and a list of variables. Line 22 shows the start of an R code block with three curly braces. The status bar at the bottom indicates the current position is 26:1 and the file is 'Actividad 1'.

```
11
12 # Actividad 1
13
14 La base de datos _hotel_ contiene información sobre reservaciones de
15 distintos hoteles. Interesa estudiar la variable target _Cancel_ la que
16 indica si la reservación fue o no cancelada.
17
18 - **Cancel**: Indica si la reserva fue cancelada o no (1: Cancelada, 0: No
19 cancelada)
20 - **Mes**: Mes de la reserva
21 - **Weekend**: Cantidad de días de fin de semana (Sábados o Domingos)
22 reservados
23 - **Weekday**: Cantidad de días de la semana (Lunes, Martes, Miércoles,
24 Jueves, Viernes) reservados
25
26
```

26:1 Actividad 1 R Markdown

# Creando un chunk

En el chunk añadimos el código que queramos entregar:

```
```{r}
library(readxl)
hotel <- read_excel("hotel.xlsx") #Carga la base de datos

#install.packages("tidyverse")
library(tidyverse)

glimpse(hotel) #Tipo de variables en la base de datos hotel
summary(hotel) #Análisis descriptivo de las variables cuantitativas
```
```

Cuando se utiliza `include=FALSE` no se muestra este código en el informe y tampoco resultados, pero sí corre.

## Actividad 1

La base de datos *hotel* contiene información sobre reservaciones de distintos hoteles. Interesa estudiar la variable *target Cancel* que indica si la reservación fue o no cancelada.

- **Cancel:** Indica si la reserva fue cancelada o no (1: Cancelada, 0: No cancelada)
- **Mes:** Mes de la reserva
- **Weekend:** Cantidad de días de fin de semana (Sábados o Domingos) reservados
- **Weekday:** Cantidad de días de la semana (Lunes, Martes, Miércoles, Jueves, Viernes) reservados

```
library(readxl)
hotel <- read_excel("hotel.xlsx") #Carga la base de datos

#install.packages("tidyverse")
library(tidyverse)

## -- Attaching packages ----- tidyverse 1.3.0 --

## v ggplot2 3.3.2      v purrr   0.3.4
## v tibble  3.0.3      v dplyr  1.0.1
## v tidyr   1.1.1      v stringr 1.4.0
## v readr   1.3.1      v forcats 0.5.0

## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

glimpse(hotel) #Tipo de variables en la base de datos hotel

## Rows: 118,987
## Columns: 4
## $ Cancel <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, ...
## $ Mes <chr> "July", "July", "July", "July", "July", "July", "July", "Ju...
## $ Weekend <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ Weekday <dbl> 0, 0, 1, 1, 2, 2, 2, 2, 3, 3, 4, 4, 4, 4, 4, 4, 4, 1, 1, 4, ...
```

# Opciones editables del chunk

- `echo=FALSE` no muestra el código pero sí el resultado
- `eval=FALSE` solo muestra el código
- `include=FALSE` no muestra el código ni el resultado, pero sí corre el código
- `message=FALSE` no muestra los mensajes producidos al correr el código en el informe
- `collapse=TRUE` elimina la separación entre una línea de código y su resultado
- `cache=TRUE` guarda los resultados de manera de no correr una y otra vez el código al compilar el informe

Para gráficos

- `fig.width` y `fig.height` modifican el tamaño del gráfico en el informe
- `fig.align='center'` ubica el gráfico al centro

# Customizando el chunk

```
```{r, echo=FALSE, message=FALSE, collapse=FALSE}
library(readxl)
hotel <- read_excel("hotel.xlsx") #Carga la base de datos

#install.packages("tidyverse")
library(tidyverse)

glimpse(hotel) #Tipo de variables en la base de datos hotel
summary(hotel) #Análisis descriptivo de las variables cuantitativas
```
```

## Actividad 1

La base de datos *hotel* contiene información sobre reservaciones de distintos hoteles. Interesa estudiar la variable target *Cancel* la que indica si la reservación fue o no cancelada.

- **Cancel:** Indica si la reserva fue cancelada o no (1: Cancelada, 0: No cancelada)
- **Mes:** Mes de la reserva
- **Weekend:** Cantidad de días de fin de semana (Sábados o Domingos) reservados
- **Weekday:** Cantidad de días de la semana (Lunes, Martes, Miércoles, Jueves, Viernes) reservados

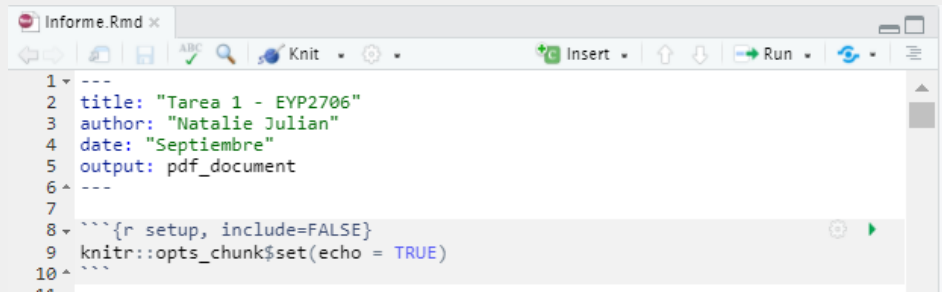
```
## Rows: 118,987
## Columns: 4
## $ Cancel <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Mes <chr> "July", "July", "July", "July", "July", "July", "July", "Ju...
## $ Weekend <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Weekday <dbl> 0, 0, 1, 1, 2, 2, 2, 2, 3, 3, 4, 4, 4, 4, 4, 4, 1, 1, 4,...

##      Cancel      Mes      Weekend      Weekday
## Min.   :0.0000 Length:118987 Min.    : 0.000 Min.    : 0.000
## 1st Qu.:0.0000 Class :character 1st Qu.: 0.000 1st Qu.: 1.000
## Median :0.0000 Mode  :character Median : 1.000 Median : 2.000
## Mean   :0.3708      Mean   : 0.927 Mean   : 2.499
## 3rd Qu.:1.0000      3rd Qu.: 2.000 3rd Qu.: 3.000
## Max.   :1.0000      Max.   :19.000 Max.   :50.000
```



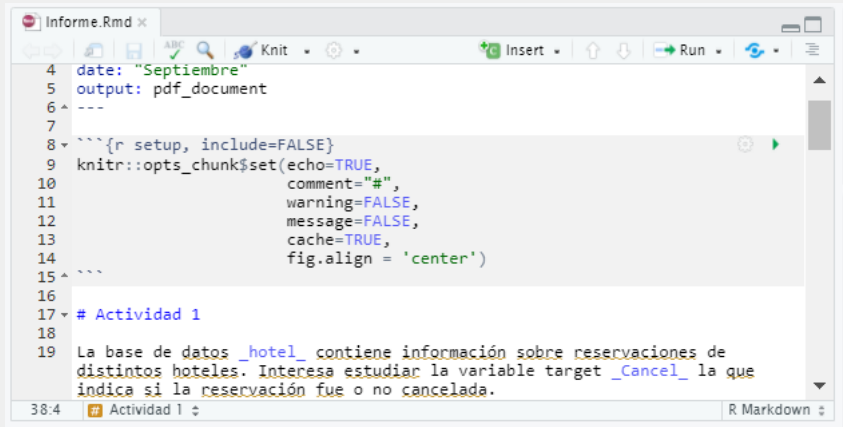
# Customizando todos los chunks

Podemos definir características de todos nuestros chunks con el chunk principal. El chunk principal es aquél que está posterior al YAML:



```
1 ---
2 title: "Tarea 1 - EYP2706"
3 author: "Natalie Julian"
4 date: "Septiembre"
5 output: pdf_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
```

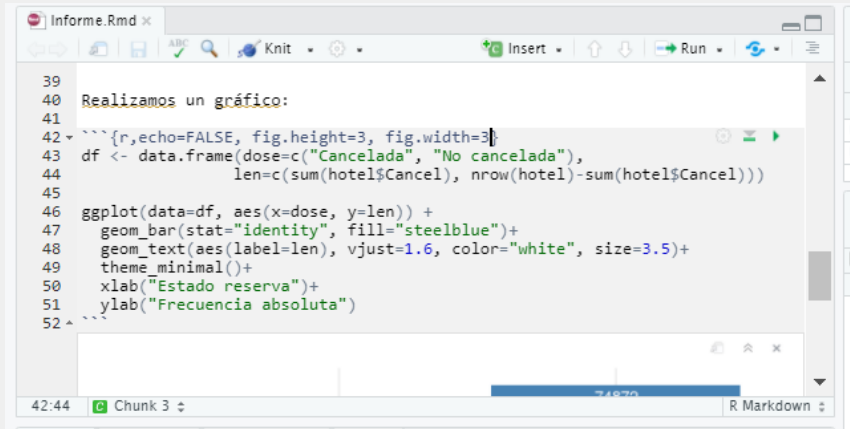
# r setup



```
Informe.Rmd x
4 date: "Septiembre"
5 output: pdf_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo=TRUE,
10                        comment="#",
11                        warning=FALSE,
12                        message=FALSE,
13                        cache=TRUE,
14                        fig.align = 'center')
15 ```
16
17 # Actividad 1
18
19 La base de datos _hotel_ contiene información sobre reservaciones de
distintos hoteles. Interesa estudiar la variable target _Cancel_ la que
indica si la reservación fue o no cancelada.

38:4 # Actividad 1 R Markdown
```


Podemos añadir un gráfico:

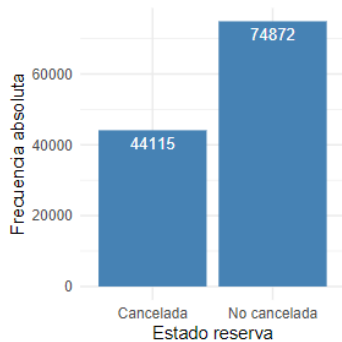


The screenshot shows an R Markdown editor window titled 'Informe.Rmd'. The code is as follows:

```
39
40 Realizamos un gráfico:
41
42 ```{r,echo=FALSE, fig.height=3, fig.width=3}
43 df <- data.frame(dose=c("Cancelada", "No cancelada"),
44                 len=c(sum(hotel$Cancel), nrow(hotel)-sum(hotel$Cancel)))
45
46 ggplot(data=df, aes(x=dose, y=len)) +
47   geom_bar(stat="identity", fill="steelblue")+
48   geom_text(aes(label=len), vjust=1.6, color="white", size=3.5)+
49   theme_minimal()+
50   xlab("Estado reserva")+
51   ylab("Frecuencia absoluta")
52 ```
```

Below the code, a partial view of a bar chart is visible. It has two bars: a short one for 'Cancelada' and a much longer one for 'No cancelada'. The y-axis is labeled 'Frecuencia absoluta' and the x-axis is labeled 'Estado reserva'. The value for 'No cancelada' is approximately 74070.

42:44 |  Chunk 3 | R Markdown



```
---
title: "Tarea 1 - EYP2706"
author: "Natalie Julian"
date: "Septiembre"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo=TRUE, comment="#", warning=FALSE, message=FALSE, cache=TRUE, fig.align = 'center')
```

# Actividad 1

La base de datos _hotel_ contiene información sobre reservaciones de distintos hoteles. Interesa estudiar la variable target _Cancel_ la que indica si la reservación fue o no cancelada.

- **Cancel**: Indica si la reserva fue cancelada o no (1: Cancelada, 0: No cancelada)
- **Mes**: Mes de la reserva
- **Weekend**: Cantidad de días de fin de semana (Sábados o Domingos) reservados
- **Weekday**: Cantidad de días de la semana (Lunes, Martes, Miércoles, Jueves, Viernes) reservados

```{r}
library(readxl)
hotel <- read_excel("hotel.xlsx") #Carga la base de datos

#install.packages("tidyverse")
library(tidyverse)

glimpse(hotel) #Tipo de variables en la base de datos hotel
summary(hotel) #Análisis descriptivo de las variables cuantitativas
```
```

Realizamos un gráfico:

```
““{r,echo=FALSE, fig.height=3, fig.width=3}
df <- data.frame(dose=c("Cancelada", "No cancelada"),
                 len=c(sum(hotel$Cancel), nrow(hotel)-sum(hotel$Cancel)))

ggplot(data=df, aes(x=dose, y=len)) +
  geom_bar(stat="identity", fill="steelblue")+
  geom_text(aes(label=len), vjust=1.6, color="white", size=3.5)+
  theme_minimal()+
  xlab("Estado reserva")+
  ylab("Frecuencia absoluta")
““
```

# Shiny Web App

Muchas veces generar un informe plano no resulta ser tan llamativo o informativo como pudiera ser un dashboard o aplicación web, en la cual, el usuario puede simular, clickear e interactuar más con nuestro trabajo.



## ui

### Interfaz del usuario

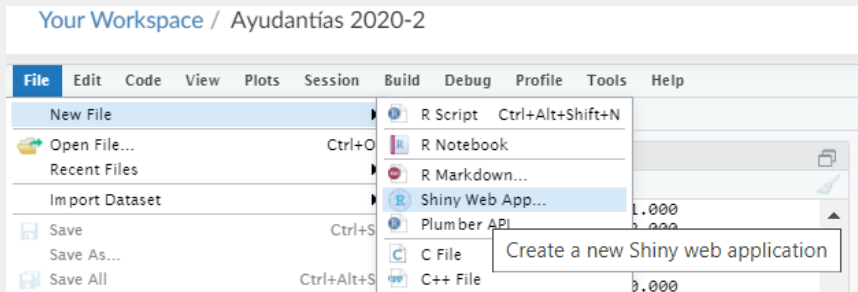
Aquí se definen todas las opciones que el usuario tendrá en la app. Por ejemplo, seleccionar categorías, seleccionar rangos, deslizarse en distintas pestañas, etcétera. Todo aquello que se le pida al usuario especificar se llamarán inputs.

## server

### Servidor


Aquí se define todo lo que continuamente corre en R. Se especifican los outputs (gráficos, tablas, resultados), todo lo que dependerá de los inputs del usuario se define aquí.

# Partiendo una Shiny app



# Partiendo una Shiny app


New Shiny Web Application



Application name:

Application type: ☒ Single File (app.R)  
☐ Multiple File (ui.R/server.R)

Create within directory:

 [Shiny Web Applications](#)

# Ejemplo

```
library(shiny)

# Define UI for application that draws a histogram
ui <- fluidPage(

  # Application title
  titlePanel("Old Faithful Geyser Data"),

  # Sidebar with a slider input for number of bins
  sidebarLayout(
    sidebarPanel(
      sliderInput("bins",
        "Number of bins:",
        min = 1,
        max = 50,
        value = 30)
    ),

    # Show a plot of the generated distribution
    mainPanel(
      plotOutput("distPlot")
    )
  )
)
```

# Ejemplo

```
# Define server logic required to draw a histogram
server <- function(input, output) {

  output$distPlot <- renderPlot({
    # generate bins based on input$bins from ui.R
    x    <- faithful[, 2]
    bins <- seq(min(x), max(x), length.out = input$bins + 1)

    # draw the histogram with the specified number of bins
    hist(x, breaks = bins, col = 'darkgray', border = 'white')
  })
}

# Run the application
shinyApp(ui = ui, server = server)
```

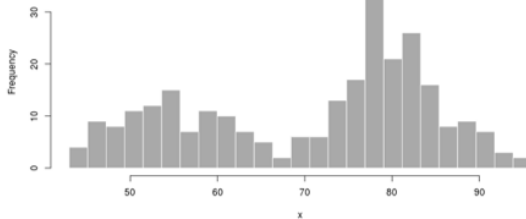
## Old Faithful Geyser Data



Paso 2: La máquina:  
En base al input, el servidor genera un output: el histograma

Paso 1: Mano del usuario: Entrega input

Histogram of x



# Desde cero generando un dashboard

```
library(shiny) #App web
library(shinydashboard) #Para formato dashboard
library(shinyjs) #Para usar entorno javascript
library(highcharter) #Para graficos interactivos
library(DT) #Para tablas
library(dplyr) #Para manipulacion de bases de datos

###Base de datos

library(readr)
Pokemon <- read_csv("Pokemon.csv")

#Barra superior del dashboard:
header <- dashboardHeader( )

#Menu de navegacion del dashboard:
sidebar <- dashboardSidebar( )

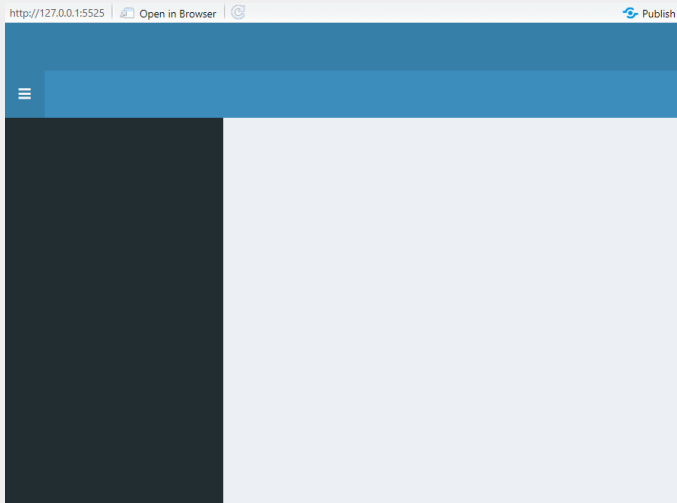
#Cuerpo de cada viñeta
body <- dashboardBody( )

ui <- dashboardPage(header, sidebar, body)

server <- function(input, output) {}

shinyApp(ui = ui, server = server)
```

Se crea un dashboard vacío:



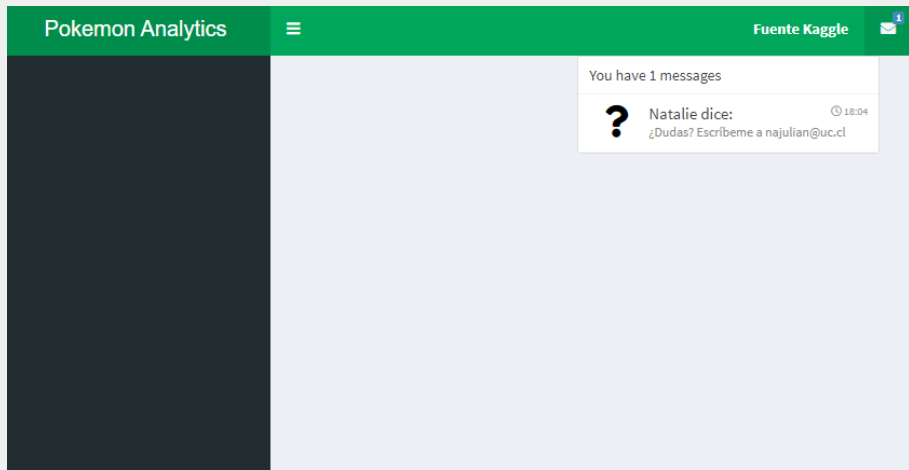


# Editando barra superior *header*

```
#Barra superior del dashboard:
header <- dashboardHeader(title = "Pokemon Analytics",
  titleWidth = 250, tags$li(
    a(strong("Fuente Kaggle"),
      height = 40,
      href = "https://www.kaggle.com/abcsds/pokemon",
      title = "Link directo"),class = "dropdown"),
    dropdownMenu(type="message", messageItem(
      from = "Natalie dice:",
      message = HTML("¿Dudas? Escribeme a najulian@uc.cl"),
      icon = icon("question"),
      time = substr(Sys.time(), start=12, stop=16)
    ))
  )
#Menu de navegacion del dashboard:
sidebar <- dashboardSidebar(width = 250)

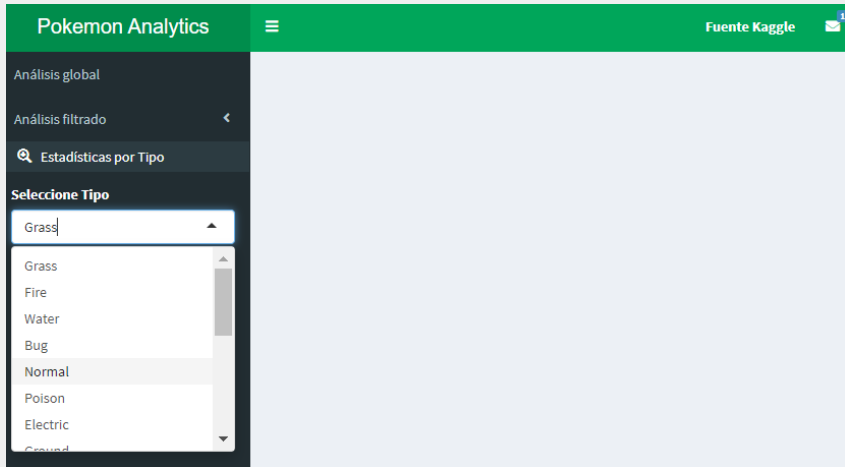
body <- dashboardBody()

#Unifica las tres partes
ui <- dashboardPage(header, sidebar, body, skin = "green")
```



# Fijando inputs en barra lateral *sidebar*

```
sidebar <- dashboardSidebar(  
  width = 250,  
  sidebarMenu(  
    id = 'sidebar',  
    style = "position: relative; overflow: visible;",  
  
    #Primera pestaña  
    menuItem("Análisis global",  
      tabName = 'menu1', startExpanded = F),  
    #Segunda pestaña  
    menuItem("Análisis filtrado", tabName = 'menu2', startExpanded = F,  
      menuSubItem('Estadísticas por Tipo',  
        tabName = "menu22",  
        icon = icon('zoom-in',  
          lib = 'glyphicon'))),  
    div(id = 'sidebar1',  
      conditionalPanel("input.sidebar === 'menu22'",  
        selectizeInput("select_region2",  
          "Seleccione Tipo",  
          choices = unique(Pokemon$'Type 1'),  
          selected = "", width = "300px",  
          multiple = F))),  
  useShinyjs()  
)  
)
```



# Orden en el *body*

```
body <- dashboardBody(  
  tabItems(  
    tabItem(tabName = 'menu1',  
            h1("Estadísticas generales"),  
            fluidRow(column(width=2, dataTableOutput('table1')))  
    ),  
    tabItem(tabName = 'menu2',  
            h1("Estadísticas filtradas por tipo"),  
            fluidRow(highchartOutput("graf1"))  
    )  
  )  
)
```

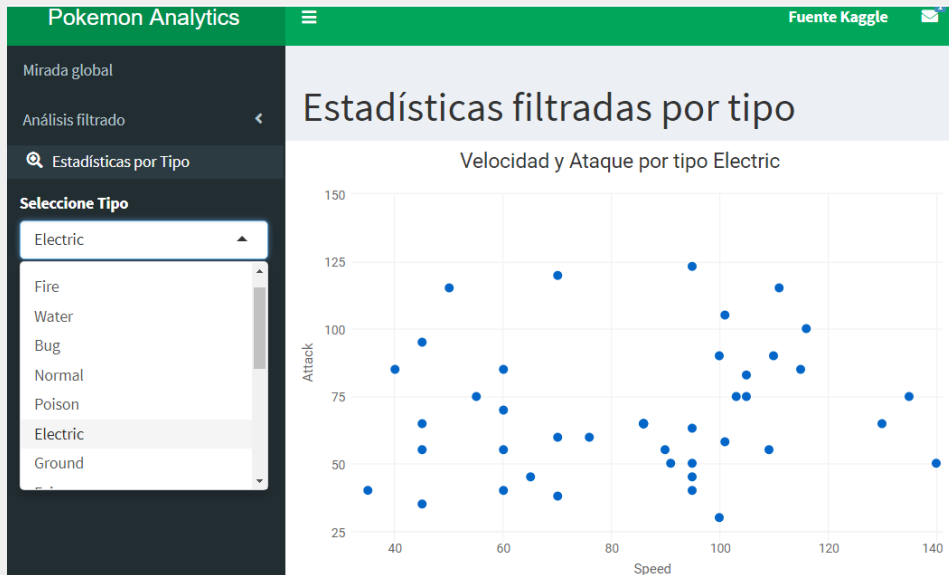
# Finalmente el server

```
#Genera outputs:
server <- function(input, output) {
  output$table1 <- renderDataTable({















    datatable(na.omit(Pokemon[,-1]), filter = list(position = "top"),
              selection="multiple",
              options = list(dom='t', autoWidth = TRUE ,
                             pageLength = 20, ordering=F,
                             searchHighlight = FALSE,
                             scrollX = FALSE,
                             class = c('compact cell-border stripe hover'),
                             columnDefs = list(list(className="dt-justify"))),
              extensions = c("Buttons" , "FixedColumns"), rownames= " ")})

  output$graf1 <- renderHighchart({
    hchart(Pokemon %>%
      filter('Type 1'==input$select_tipo1),
      "scatter", hcaes(x = Speed, y = Attack)) %>%
    hc_yAxis(title = list(text = "Attack"))%>%
    hc_title(text=paste("Velocidad y Ataque por tipo", input$select_tipo1), align = "center")%>%
    hc_tooltip(pointFormat= "Attack: {point.y} <br>
      Speed:{point.x}" ) %>%
    hc_add_theme(hc_theme_google())

  })
}
```

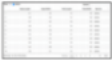



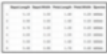



# Más inputs

|   |   |  |  |
|---|---|--|--|
|  | <b>actionButton</b> (inputId, label, icon, ...)   |  | <b>numericInput</b> (inputId, label, value, min, max, step)  |
|  | <b>actionLink</b> (inputId, label, icon, ...)   |  | <b>passwordInput</b> (inputId, label, value)   |
|  | <b>checkboxGroupInput</b> (inputId, label, choices, selected, inline)   |  | <b>radioButtons</b> (inputId, label, choices, selected, inline)  |
|  | <b>checkboxInput</b> (inputId, label, value)  |  | <b>selectInput</b> (inputId, label, choices, selected, multiple, selectize, width, size) (also <b>selectizeInput</b> ()) |
|  | <b>dateInput</b> (inputId, label, value, min, max, format, startview, weekstart, language)                      |  | <b>sliderInput</b> (inputId, label, min, max, value, step, round, format, locale, ticks, animate, width, sep, pre, post) |
|  | <b>dateRangeInput</b> (inputId, label, start, end, min, max, format, startview, weekstart, language, separator) |  | <b>submitButton</b> (text, icon)<br>(Prevents reactions across entire app)   |
|  | <b>fileInput</b> (inputId, label, multiple, accept)   |  | <b>textInput</b> (inputId, label, value)   |



# Más outputs

|   |   |            |   |
|---|---|------------|---|
|  | <b>DT::renderDataTable</b> (expr, options, callback, escape, env, quoted) | works with | <b>dataTableOutput</b> (outputId, icon, ...)  |
|  | <b>renderImage</b> (expr, env, quoted, deleteFile)                        |            | <b>imageOutput</b> (outputId, width, height, click, dblclick, hover, hoverDelay, hoverDelayType, brush, clickId, hoverId, inline) |
|  | <b>renderPlot</b> (expr, width, height, res, ..., env, quoted, func)      |            | <b>plotOutput</b> (outputId, width, height, click, dblclick, hover, hoverDelay, hoverDelayType, brush, clickId, hoverId, inline)  |
|  | <b>renderPrint</b> (expr, env, quoted, func, width)                       |            | <b>verbatimTextOutput</b> (outputId)  |
|  | <b>renderTable</b> (expr, ..., env, quoted, func)                         |            | <b>tableOutput</b> (outputId)   |
| <b>foo</b>  | <b>renderText</b> (expr, env, quoted, func)                               |            | <b>textOutput</b> (outputId, container, inline)   |
|  | <b>renderUI</b> (expr, env, quoted, func)                                 |            | <b>uiOutput</b> (outputId, inline, container, ...)<br>& <b>htmlOutput</b> (outputId, inline, container, ...)                      |