

Tablas y graficos con R y R Studio

Roberto Gil Saura

Última actualización: 2021-06-06

Contents



Prefacio

Todavía recuerdo cuando hace unos años descubrí R (?) -lejos queda ya aquel 2000- y me pareció una herramienta valiosa para mi trabajo, pero que no era para mí. La percibí para usuarios programadores y no para analistas o científicos de datos, como gusta llamar hoy en día a estos trabajadores del dato. Sin embargo, hace tres años (2018) y tras una etapa en la que centré mis energías y esfuerzos en mi PhD, redescubrí esta herramienta y no sólo ella, también R Studio y todo el trabajo que se había realizado en estos últimos años. Quedé gratamente impresionado y me puse manos a la obra porque vi un enorme potencial en la misma para lo que era mi trabajo actual. Una de las cuestiones fundamentales que aprecié es que las dos herramientas ya no sólo se enfocaban (probablemente nunca lo habían hecho) sólo a los programadores, sino que se apreciaba mucho el enfoque a analistas con todo tipo de formación y con escasos conocimientos de programación, pero que no temían enfrentarse a algo que les iba a reportar eficiencia en su trabajo y un enorme potencial de evolución. Con el paso del tiempo fui descubriendo todo lo que R y R Studio podrían hacer por mí (y por mis alumnos), y lo que ganaba / ahorra en mi trabajo y en las soluciones que podía ofrecer a mis colegas y/o clientes. Y ahí empezó todo.

En este documento, sólo pongo el foco en dos aspectos fundamentales en el mundo de la investigación de mercados cuantitativa: las tablas de contingencia y los gráficos, en sus diferentes modalidades y con sus diferentes sabores, básico y avanzados, visuales y no tan visuales; todo ello condimentado y aderezado con una cantidad de paquetes suplementarios y la inestimable ayuda de esos programadores que desinteresadamente ceden su trabajo a la comunidad. En especial aquí debo agradecer a Gregory Demin, Joshua Kunst y a Yihui Xie su desinteresada colaboración en la creación de diferentes paquetes y utilidades que han servido para poder plasmar toda la información que aquí se desarrolla.

Cita

Gil-Saura, R., 2021. Tablas y gráficos con R y R Studio. 1st ed. [ebook] València: InvestigaOnline.com. ISBN: 978-84-09-29382-7; disponible en: <https://tables.investigaonline.com>.

Licencia

© Textos y gráficos: Roberto Gil-Saura (robertogil@investigaonline.com)

Cualquier forma de reproducción, distribución, comunicación pública o transformación de esta obra solo podrá ser realizada con la autorización expresa del autor bajo los términos abajo descritos.

1ª edición: Valencia, 2021 ISBN: 978-84-09-29382-7 Última actualización: 2021-06-06

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 Unported License (CC BY-NC-SA 4.0) <http://creativecommons.org/licenses/by-nc/4.0/> In short: Roberto Gil-Saura retains the Copyright but you are free to reproduce, reblog, remix and modify the content only under the same license to this one. You may not use this work for commercial purposes but permission to use this material in nonprofit teaching is still granted, provided the authorship and licensing information here is displayed.



Chapter 1

Introducción

1.1 ¿Por qué este libro?

La respuesta más simple a la pregunta ¿Por qué este libro? es que no hay libros dedicados a ilustrar y mostrar cómo hacer tablas de contingencia y gráficos a partir de esas tablas usando R; hablando más específicamente usando **expss** y **highcharter**. Por otro lado, a partir de mi experiencia y de muchos años dedicado al mundo de la investigación y la docencia (mi hobby personal) encontré el tiempo y la motivación para escribirlo. Quería crear un recurso que fuera diferente del resto de textos con el mismo contenido aproximado y porque creo que para que las ideas se difundan y adopten ampliamente, deben percibirse de manera amigable y fácil de entender, lo que tiene por qué estar reñido con una baja calidad científica y teórica. Si algo se percibe como complejo e intimidante, solo pocos creerán que ese recurso es apropiado para ellos. Así pues, y aunque todos los análisis requieren las habilidades propias de una persona acostumbrada a manejar fuentes de datos variadas, creo realmente que el recurso es extremadamente sencillo, funcional, comprensible y válido para todos.

¿No es acaso necesario y nos viene muy bien el poder utilizar macros en software como Microsoft Office® o sintaxis en IBM SPSS® ? Pues esta es la respuesta, mostrar de una forma sencilla la posibilidad de que el usuario pueda utilizar código R (?) en sus análisis de datos, que le permitan ir mucho más allá de lo que ofrecen las herramientas básicas comerciales.

Además, la elección de R no es casual:

- La primera razón para elegir R es que es un programa extremadamente potente para manipular y analizar datos. Su popularidad en alza lo ha convertido en el software de referencia para estadísticas y análisis en muchas disciplinas, y se enseña cada vez más en muchas universidades. R se ha

convertido en uno de los estándares de facto de la industria del proceso, análisis y visualización de datos, dentro de ese nuevo ámbito que es al analítica o ciencia de datos, que está dando cabida a muchos profesionales de nueva creación y en la que nos estamos integrando muchos otros que provenimos de ámbitos diferentes.

- La segunda razón para seleccionar R es que es independiente de la plataforma (puede usarla en Windows®, Mac o Unix) y es gratis y por ello no se deb invertir un euro en licencias, mantenimientos o actualizaciones. Los proveedores de software comercial pueden brindar soporte y ofrecer algún tipo de garantía, pero eso es secundario. No se puede superar el precio y la funcionalidad de R. R se enriquece por el hecho de que muchas personas de todo el mundo contribuyen y comparten su propio trabajo en forma de funcionalidades incluidas en paquetes de lo más variado y que llegan al mercado con una celeridad imposible para cualquier otro software comercial. Asimismo, R tiene recursos de ayuda inigualables, tanto en línea como físicos. Hay muchos foros en línea, listas de correo electrónico, grupos de interés, foros, blogs y sitios web llenos de información rica sobre R. Además, cada año se publican cada vez más muchos libros de R de excelente calidad. Lo que realmente se valora de R, es su carácter de código abierto, lo que permite (si sabes) ver el código por dentro alejándose de ser una caja negra.

1.2 ¿Para quién este libro?

El presente documento enfoca e introduce al usuario de investigación (fundamentalmente de Investigación de Mercados) en la creación de información analítica y reproducible utilizando scripting con *Markdown*, que permita obtener cualquier tipo de tabla resumen de datos o cualquier análisis que pueda hacer utilizando código R.

Este manual ha sido redactado pensando en los usuarios que trabajan de forma conjunta con R Studio y R. Por tanto, se asume que el usuario estará familiarizado con el uso de ambas aplicaciones y se le supone conocimiento básico de cómo funcionar con ellas. No obstante, al inicio de la sección 3, el usuario puede encontrar unos sencillos rudimentos de como trabajar para iniciarse con el trabajo de R Studio y por extensión de R.

1.3 Estructura del documento

En investigación de mercado, las tablas de contingencia o cruzadas probablemente sean el análisis de datos más predominante de todos los utilizados, complementado por una potente visualización en forma de gráfico de esos datos.

Se pretende dotar al usuario de de las herramientas básicas que le permitan reproducir cualquier tipo de cuadro o tabla, de tipo marginal o cruzada, con medidas estadísticas básicas, combinando estadísticos y frecuencias o porcentajes, o realizar pruebas de significación estándar, así como la reproducción de los gráficos que se adaptarían a esas tablas. La estructura del manual se adapta por tanto a este objetivo, organizándose de la siguiente forma:

- esta introducción al manual, donde también se ayuda a la instalación de ambas herramientas de trabajo R y R Studio;
- una sección que dedicamos a introducir al lector en el lenguaje R; no se pretende hacer un revisión exhaustiva del lenguaje, sino simplemente enumerar los términos y conceptos que serán necesarios para entender el funcionamiento de las tablas y el manejo y gestión de las mismas;
- una sección donde se introduce al usuario en la escritura de sencillas órdenes que como resultado obtendrán un cuadro o tabla;
- una sección dedicada a la creación de las tablas cruzadas;
- una sección dedicada a tablas más especiales, que denominaremos multi dimensionales;
- una sección dedicada a la generación de otras tablas auxiliares: cuadros, escalas, etc.;
- un epígrafe que describirá las pruebas básicas de significación que se utilizan en las tablas de contingencia;
- una sección dedicada a los gráficos;
- un epígrafe dedicado a operaciones con tablas, entre filas y/o columnas y a mejorar la visualización de las tablas y obtener gráficos a partir de estas operaciones especiales, ampliando el conjunto de paquetes a utilizar y combinando sus funcionalidades;
- por último, una sección complementaria más de lenguaje R, para saber cómo se incluyen las condiciones para cualquier tipo de selección de datos.

Para seguir este manual, tal y como hemos comentado anteriormente, asumimos que el usuario conoce mínimamente R y R Studio y que ha constatado su necesidad de reproducir tipos de tabla que ha visto que son posibles de obtener en Quantum, Minitab, Systat, Star, SPSS o BarbWin entre otros. Explicaremos los rudimentos básicos de trabajo en R, pero adecuados a su uso en R Studio, con una primera introducción a cómo integramos R en su flujo de trabajo.

Agradecemos expresamente a Gregory ?, desarrollador del paquete **expss** de R por su desinteresado trabajo en código abierto que permite reproducir e ir más allá de los cuadros resumen creados; al igual que a Joshua ?, desarrollador del paquete **highcharter** por su excelente aportación en el mundo de la graficación con la creación del *wrapper* para la librería Highchart, renombrada y conocida librería de gráficos JavaScript que permite su uso en R. Y no puedo acabar esta sección sin mencionar al equipo de R Studio y en particular a Yihui ?, que con sus diferentes aportaciones, permite que esta obra llegue a vuestros ojos en las condiciones que lo hace usando el paquete **bookdown** desde R Studio.

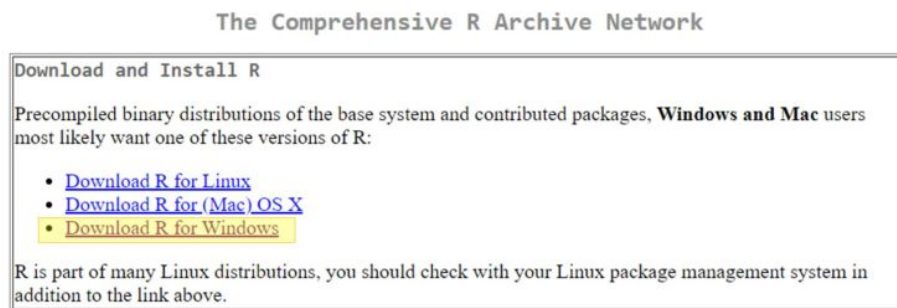
1.4 Instalación de R y R Studio

Para poder seguir el manual de forma correcta, deberemos tener instalados R y R Studio. Sigue estos pasos para poder hacerlo en Windows. Si es sobre Linux o MacOS, busca información sobre cómo instalarlo específicamente en tu versión. Es muy sencillo, pero con unas imágenes se ve mejor.

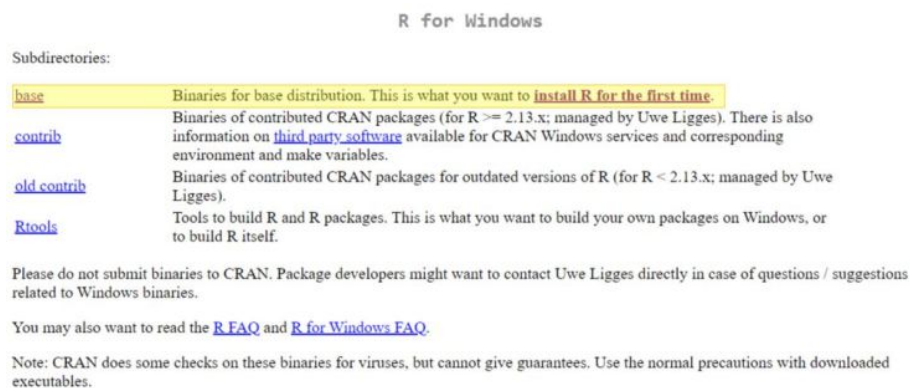
1.4.1 Instalación de R en Windows

Los pasos son los siguientes:

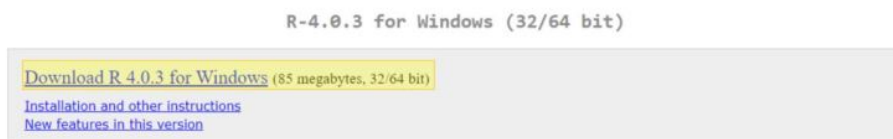
1. Abre el siguiente enlace: <https://cran.r-project.org/>
2. Selecciona la opción “Download R for Windows”.



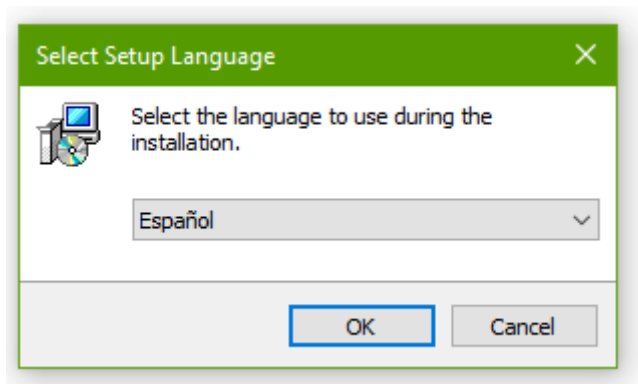
3. En la nueva página seleccione “base”.



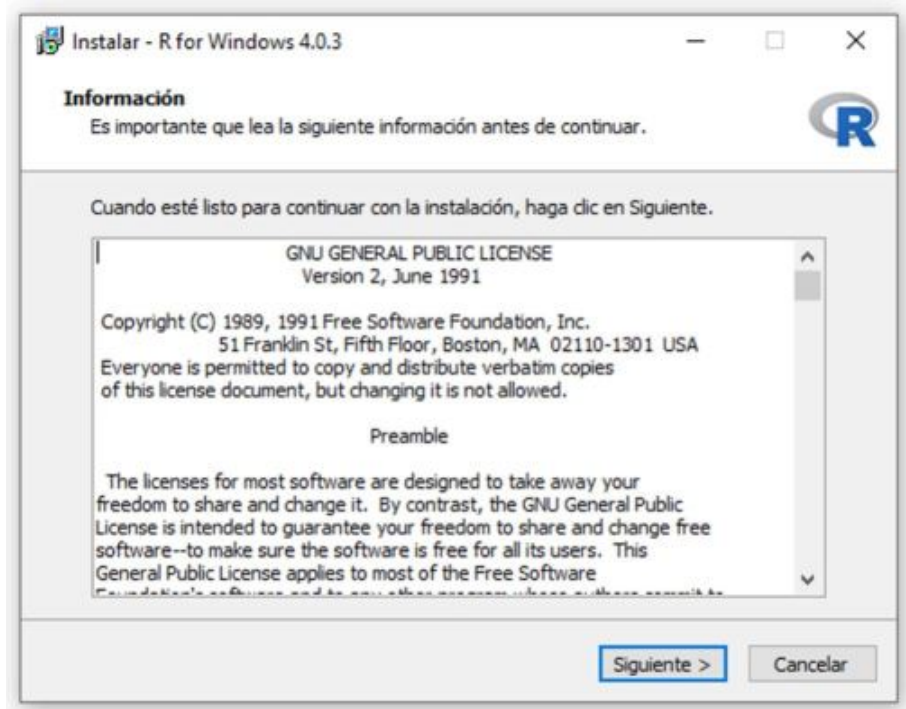
4. En la nueva página presiona en “Download R X.X.X for Windows”, donde X.X.X corresponde a la versión más actualizada disponible en ese momento. Para este caso es 4.0.3.



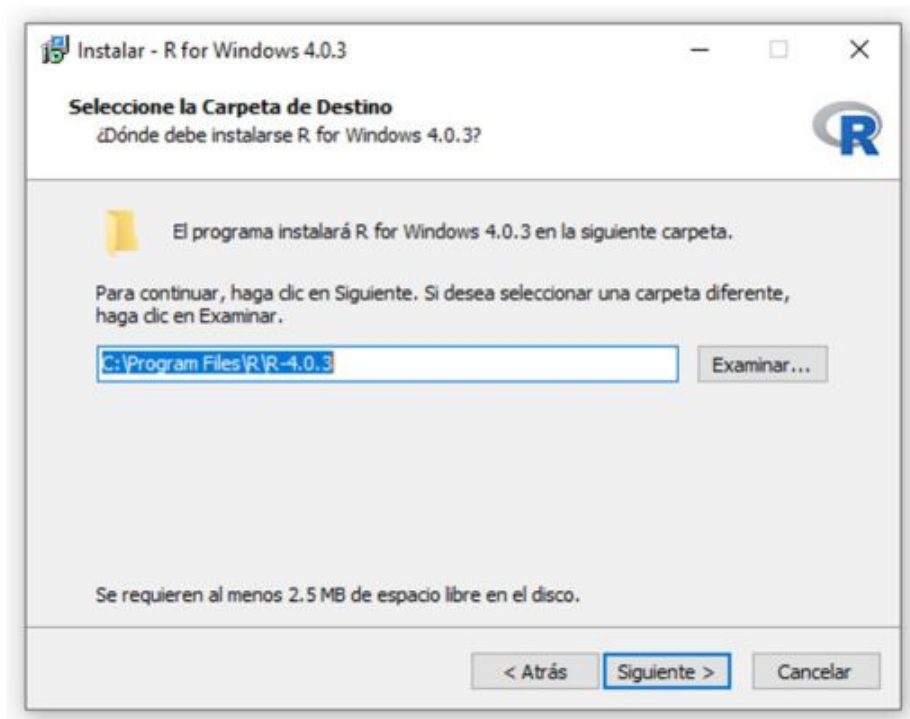
5. El archivo instalador empieza a descargarse como cualquier otro documento, la ubicación de la descarga y la forma en que se realice dependerán de la configuración que estés usando en tu navegador de internet.
6. Una vez que la descarga se complete, ejecuta el archivo de instalación desde la carpeta donde se haya almacenado.
7. Selecciona el idioma de preferencia y presiona OK.



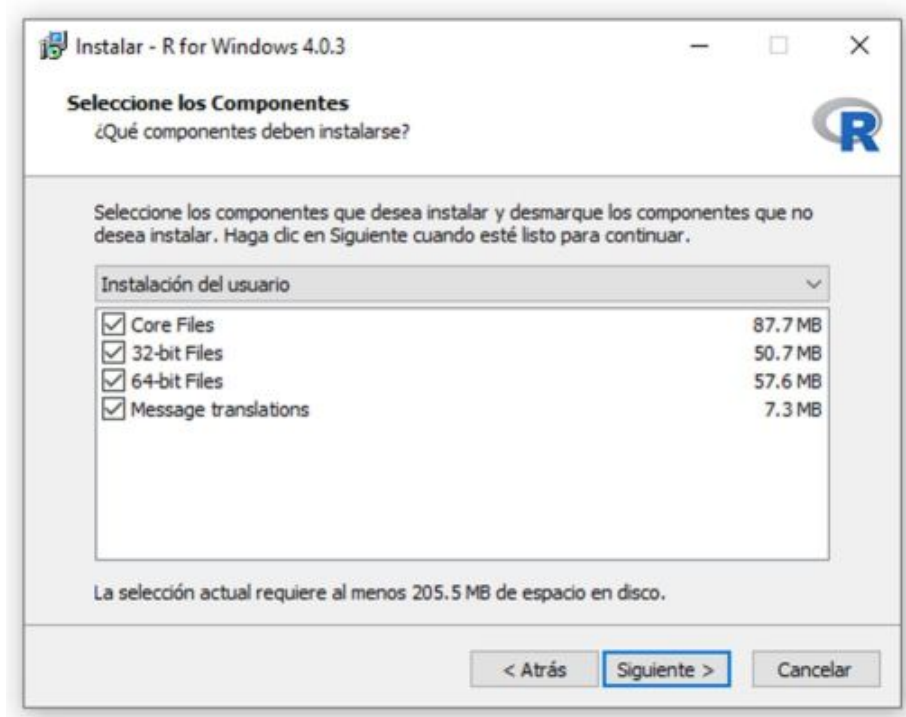
8. Sigue las recomendaciones de la pantalla y luego presiona "Siguiente".



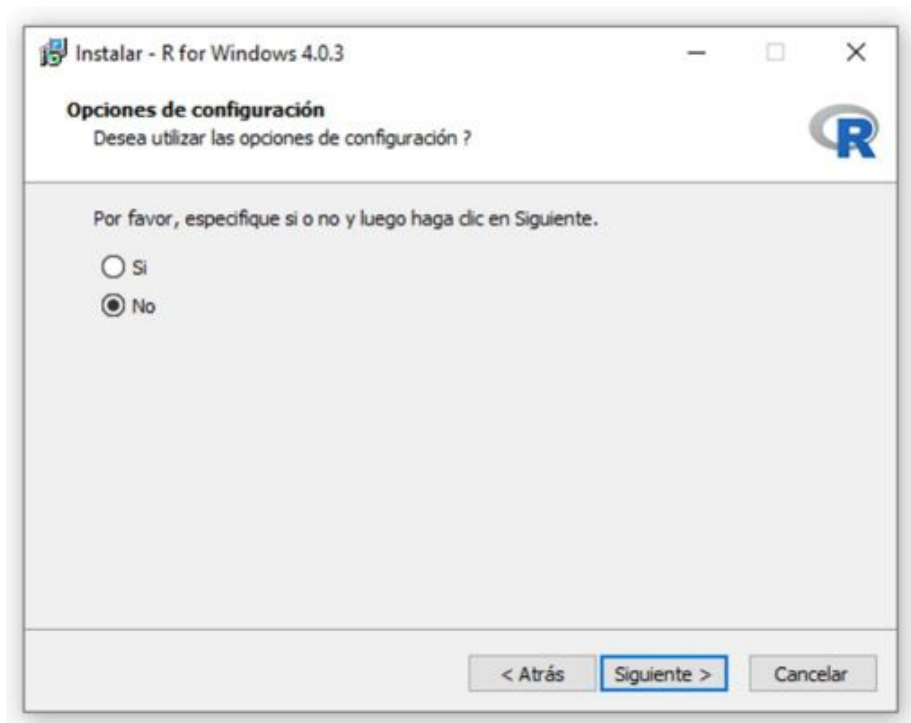
9. Selecciona la carpeta donde quieres instalar R. La recomendación es dejar la carpeta que viene por defecto. Luego, presiona “Siguiente”.



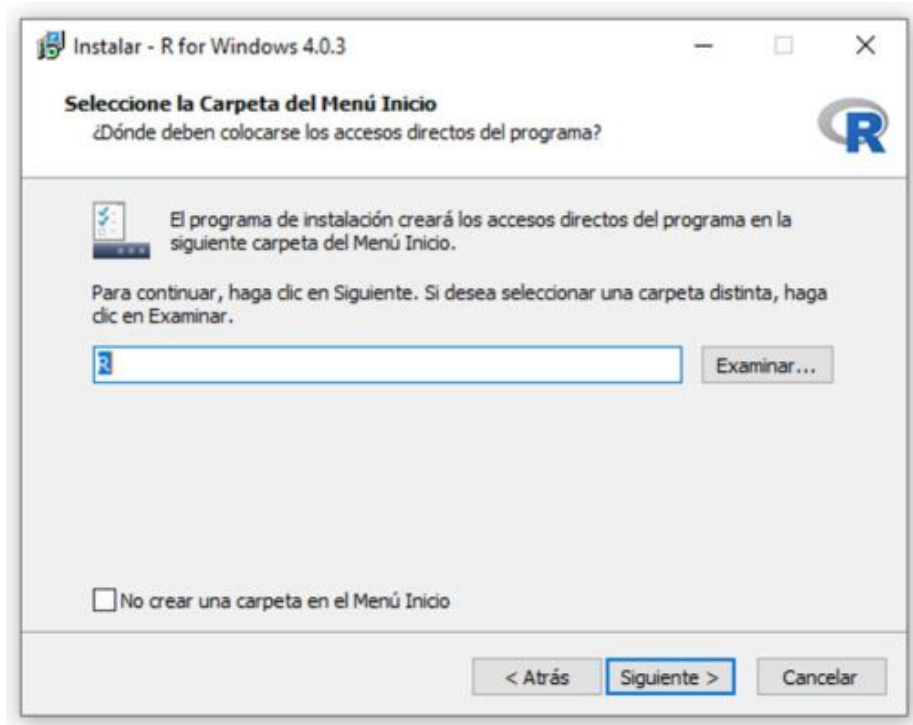
10. Selecciona los componentes que desea instalar. Es recomendable dejar seleccionados los que vienen por defecto. Luego, presiona "Siguiente".



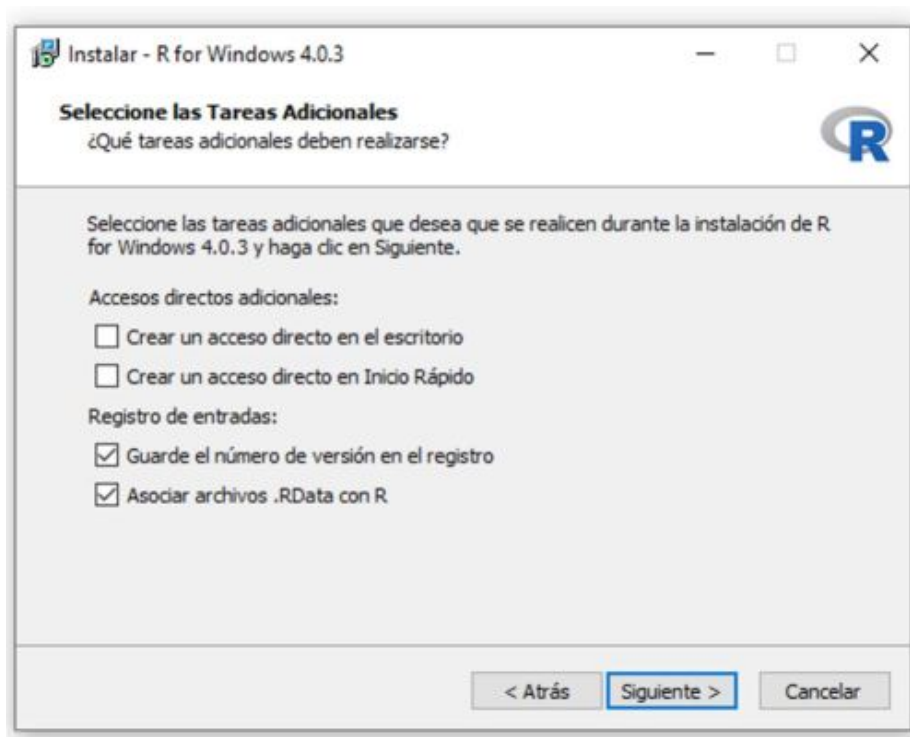
11. Especifica las opciones de configuración. En este caso, como es la instalación básica, la opción sugerida es "No". Luego, presiona "Siguiente".



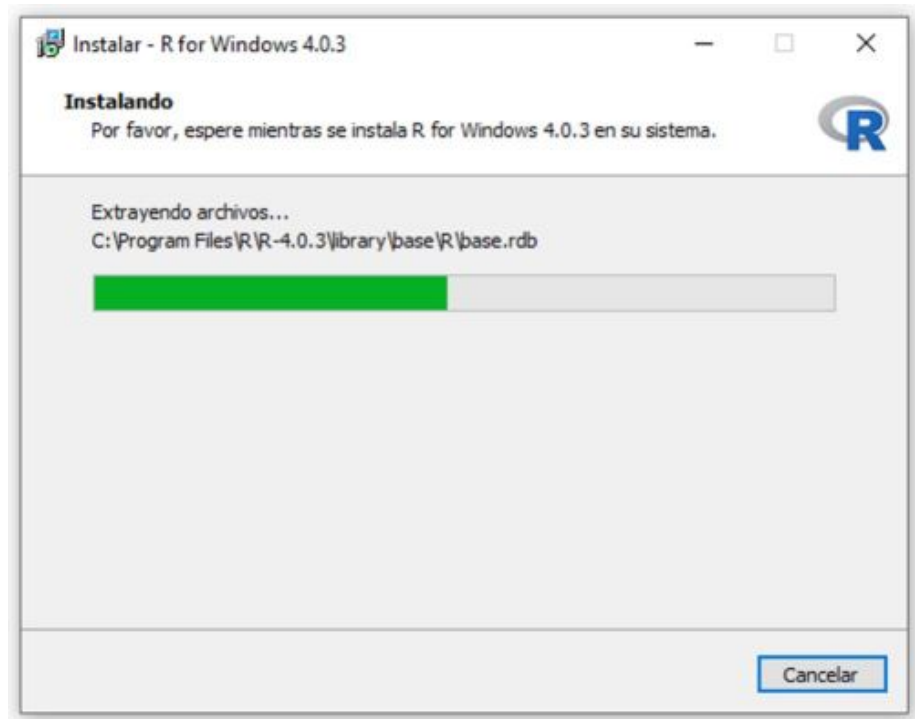
12. Lee las indicaciones y selecciona las opciones de su conveniencia. La sugerencia es dejar las opciones que vienen por defecto. Luego, presiona "Siguiete".



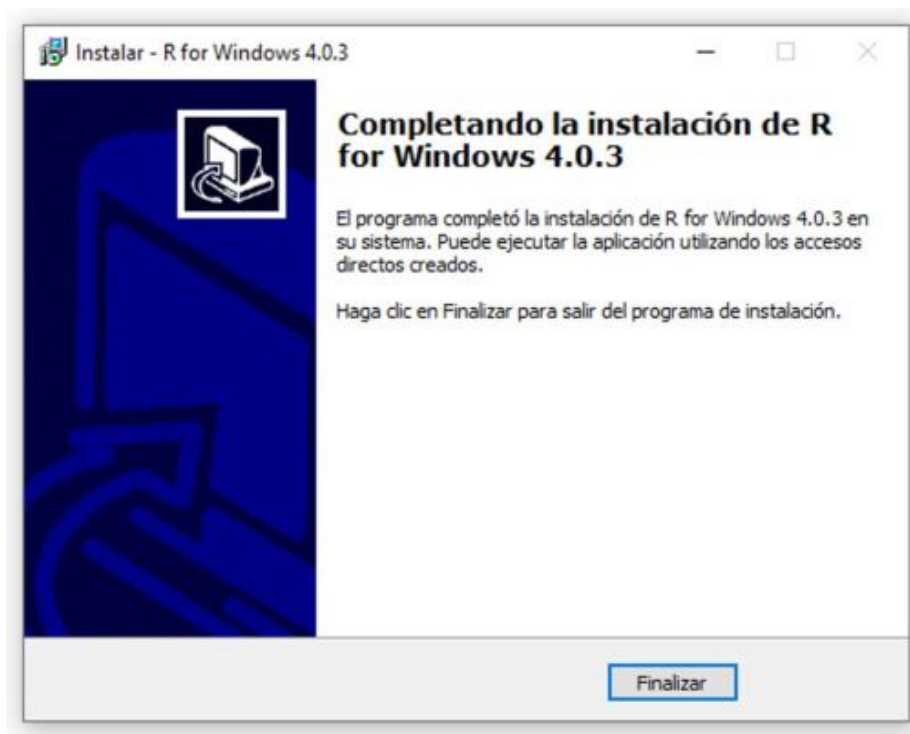
13. Selecciona las tareas adicionales. La sugerencia es dejar habilitadas solamente las que vienen por defecto. Luego, presiona "Siguiente".



14. Esto iniciará el proceso de instalación. No cierras la ventana hasta que el programa indique que el proceso ha sido completado con éxito.



15. Esto completa la instalación de R. Presiona “Finalizar”.



1.4.2 Instalación de R Studio en Windows

Vamos ahora a proceder con R Studio.

1. Abre el siguiente enlace <https://RStudio.com/products/RStudio/download/#download>
2. En esta página, se indica lo siguiente:
 - 2.1. Install R: Se requiere que R esté preinstalado. La versión más antigua soportada es 3.0.1, si se siguieron los pasos descritos anteriormente para la instalación de R tendríamos la versión más nueva, por lo que no tendríamos problema.
 - 2.2. Download R Studio Desktop: R Studio provee un instalador sugerido basado en su sistema. En la imagen de abajo el instalador sugerido es Windows porque es el sistema operativo detectado. Si el instalador coincide con su sistema operativo, haga click en el recuadro para iniciar la descarga. Si requiere otro instalador, mirar el punto 3.

RStudio Desktop 1.4.1106 - [Release Notes](#)

1. Install R. RStudio requires [R 3.0.1+](#).
2. Download RStudio Desktop. Recommended for your system:



Requires Windows 10/8/7 (64-bit)



3. En caso de requerir instaladores para otros sistemas operativos, estos se pueden encontrar en el cuadro abajo de la imagen en la misma página “All Installers”.

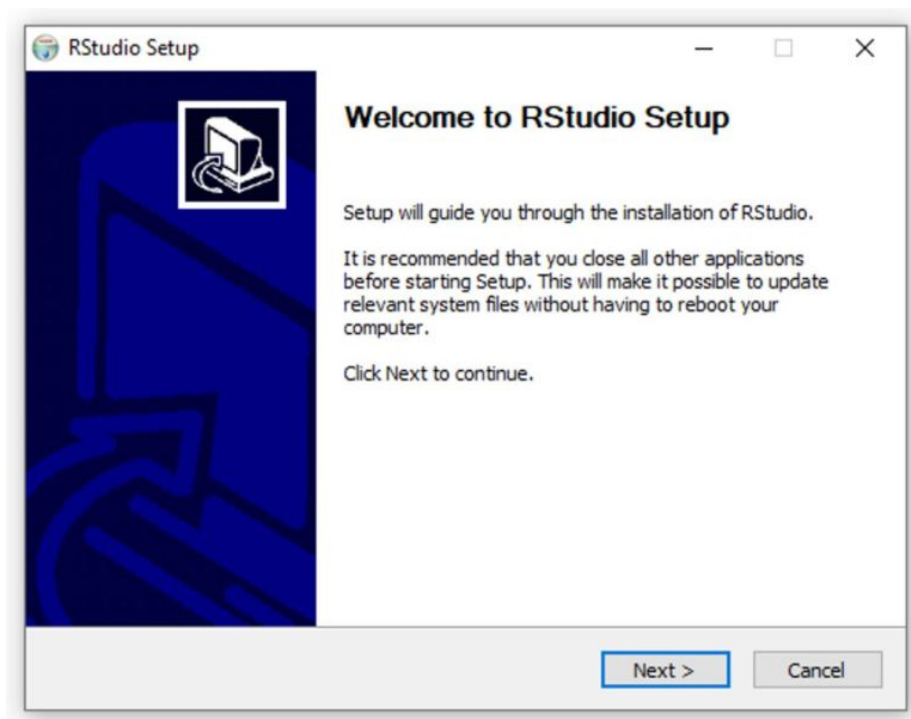
All Installers

Linux users may need to [import RStudio's public code-signing key](#) prior to installation, depending on the operating system's security policy.

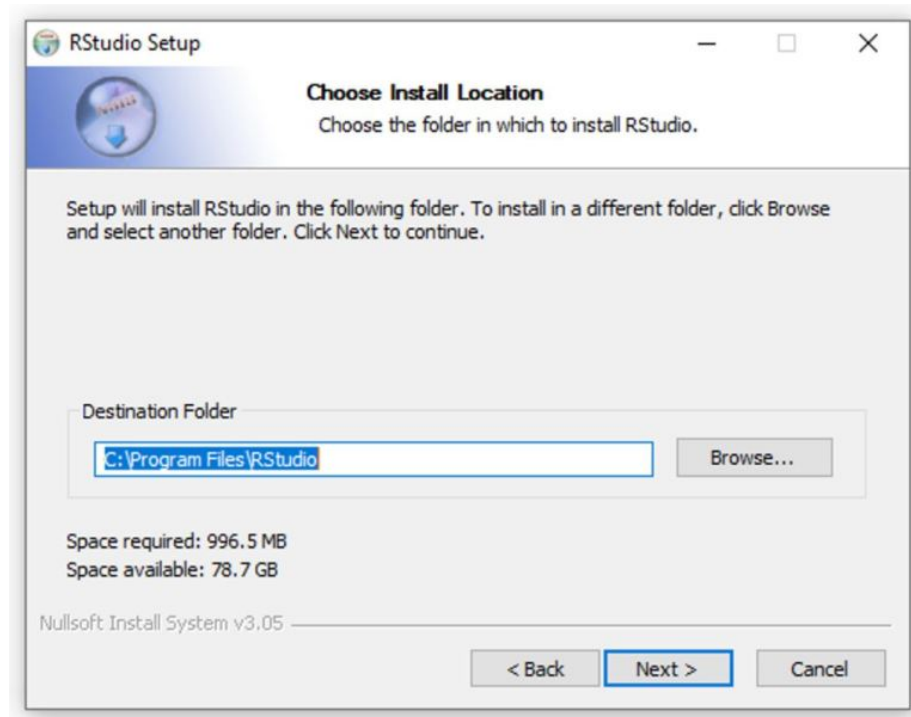
RStudio requires a 64-bit operating system. If you are on a 32 bit system, you can use an [older version of RStudio](#).

OS	Download	Size	SHA-256
Windows 10/8/7	RStudio-1.4.1106.exe	155.97 MB	d2ff8453
macOS 10.13+	RStudio-1.4.1106.dmg	153.35 MB	c64d2cda
Ubuntu 16	rstudio-1.4.1106-amd64.deb	118.45 MB	1fc82387
Ubuntu 18/Debian 10	rstudio-1.4.1106-amd64.deb	121.07 MB	3b5d3835
Fedora 19/Red Hat 7	rstudio-1.4.1106-x86_64.rpm	138.18 MB	a9e6ddc4
Fedora 28/Red Hat 8	rstudio-1.4.1106-x86_64.rpm	138.16 MB	35e57c1c
Debian 9	rstudio-1.4.1106-amd64.deb	121.33 MB	c7c9dd68
OpenSUSE 15	rstudio-1.4.1106-x86_64.rpm	123.57 MB	3539d9c3

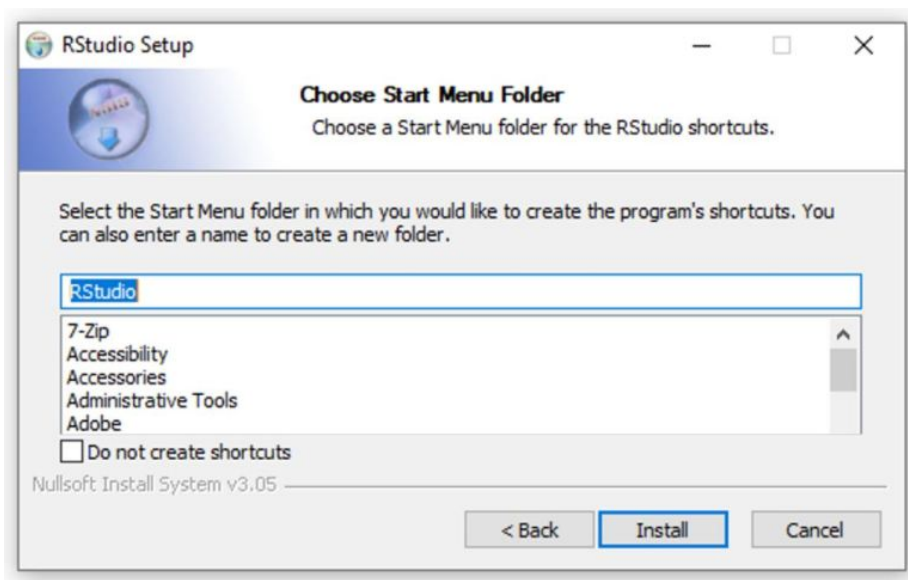
4. Después de seleccionar la descarga del instalador correspondiente, el archivo empezará a descargarse como cualquier otro documento, la ubicación de la descarga y la forma en que se realice dependerán de la configuración que estés usando tu navegador de internet.
5. Una vez que la descarga se complete, ejecuta el archivo de instalación desde la carpeta donde haya sido descargado.
6. La ventana de bienvenida indica que es recomendado cerrar todas las demás aplicaciones que se estén usando antes de iniciar la instalación. Esto es para que sea posible actualizar archivos importantes en el sistema sin necesidad de reiniciar el PC. Presiona “Next” cuando se haya completado lo anterior o si no hay inconveniente con reiniciar el PC.



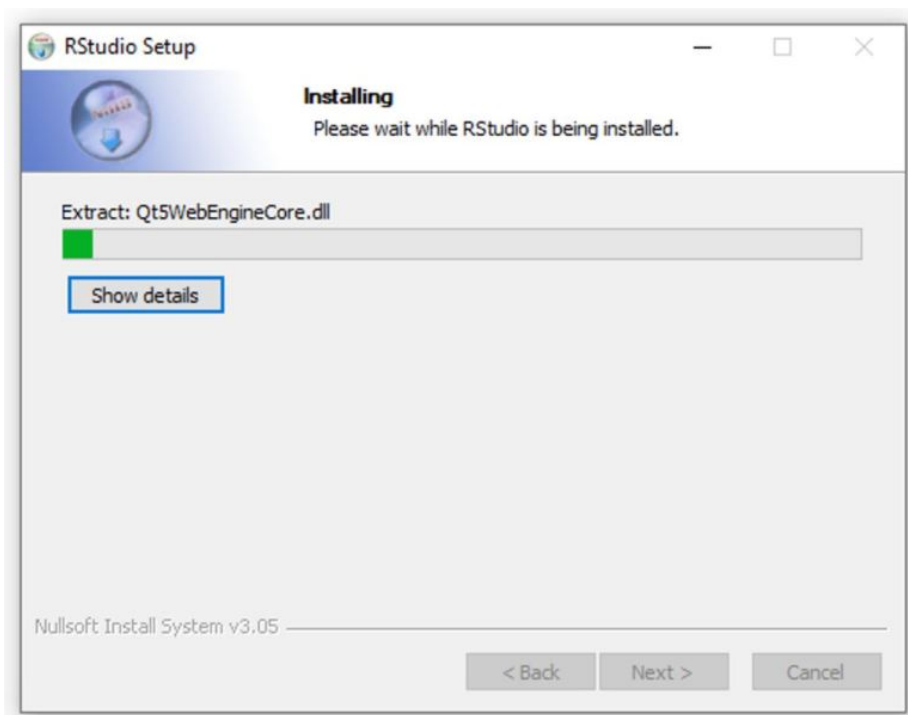
7. Selecciona la carpeta donde instalar R Studio. La recomendación es dejar la carpeta que viene por defecto. Luego, presiona "Next".



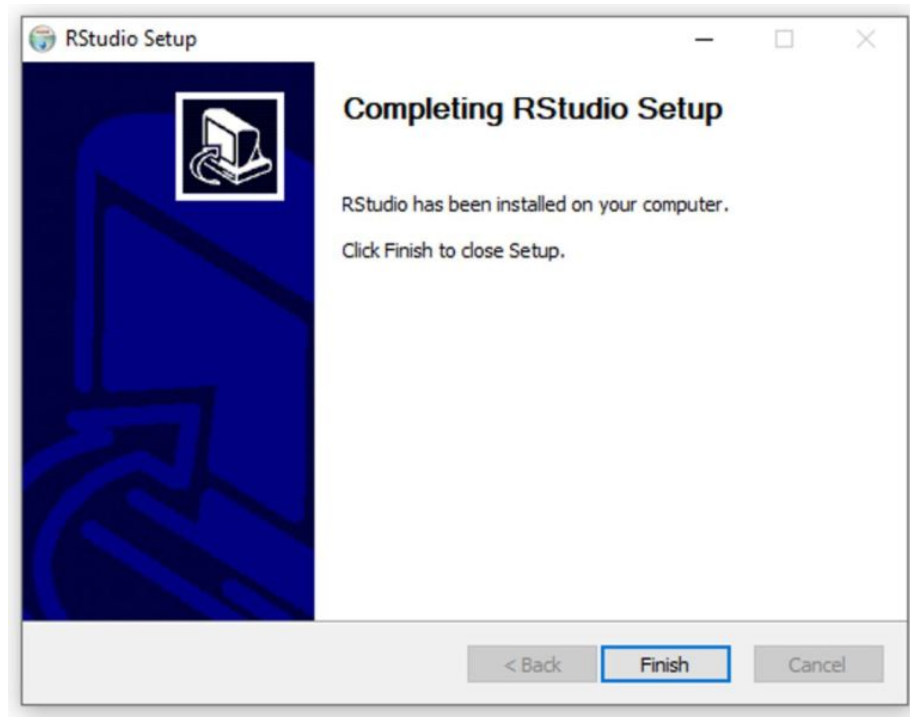
8. Selecciona la carpeta del menú de Inicio en el que se crearán los accesos directos al programa, o escribe un nombre para crear una nueva carpeta. La sugerencia es dejar las opciones que vienen por defecto. Luego, presiona “Install”.



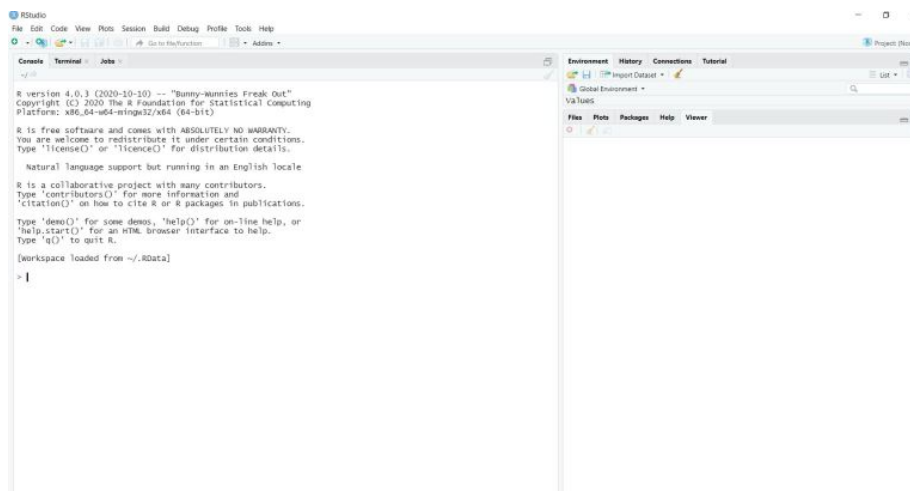
9. Esto iniciará el proceso de instalación. No cierras la ventana hasta que el programa indique que el proceso ha sido completado con éxito.



10. Esto completa la instalación de R Studio. Presiona “Finalizar”; es posible que debas reiniciar el PC dependiendo de lo establecido en el paso 6.



11. Tras el posible reinicio o sin el reinicio, lanza R Studio como cualquier otra aplicación. este es su aspecto.



A partir de aquí continuamos. R Studio se encarga de gestionar R, no deberemos preocuparnos por éste salvo para actualizarlo cuando creamos necesario o tengamos alguna necesidad puntual. R y R Studio son como cualquier otra aplicación de Windows, harás lo mismo que con las demás.

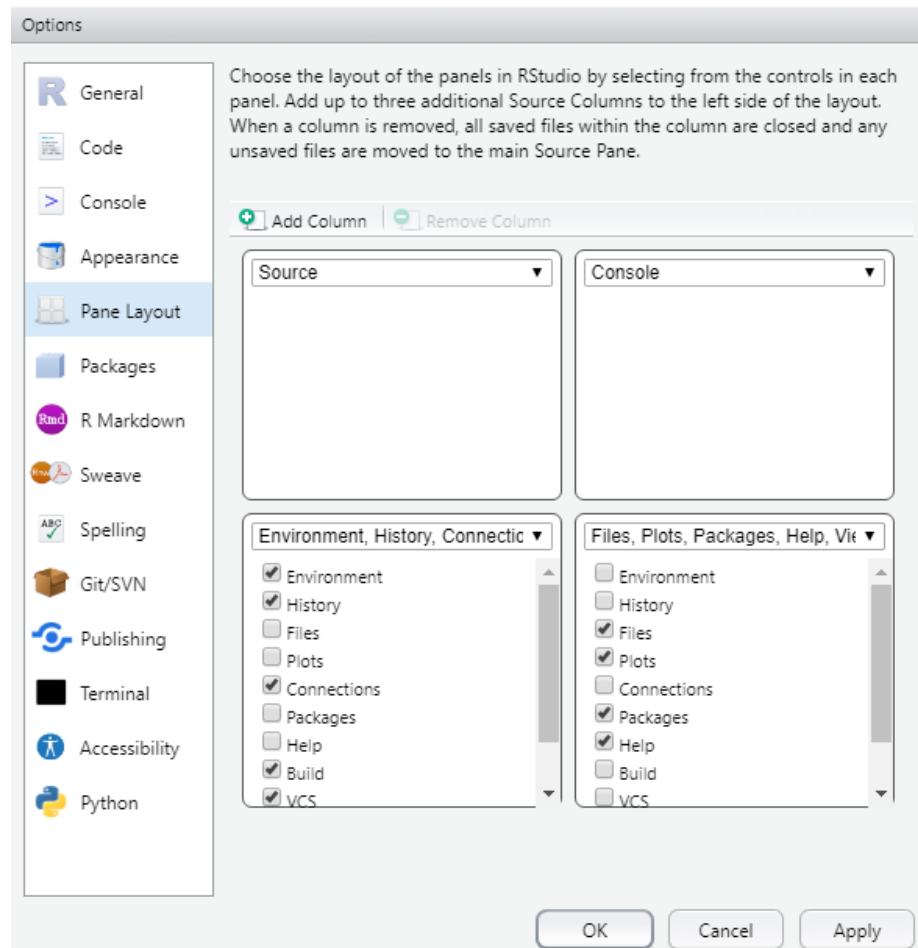
1.5 Uso básico de R Studio y de R

Si eres un lector / usuario que ya ha trabajado con R y R Studio, puedes saltar ya lo que resta de sección y pasar directamente a la sección 3. Si no lo eres, unas breves instrucciones para comenzar.

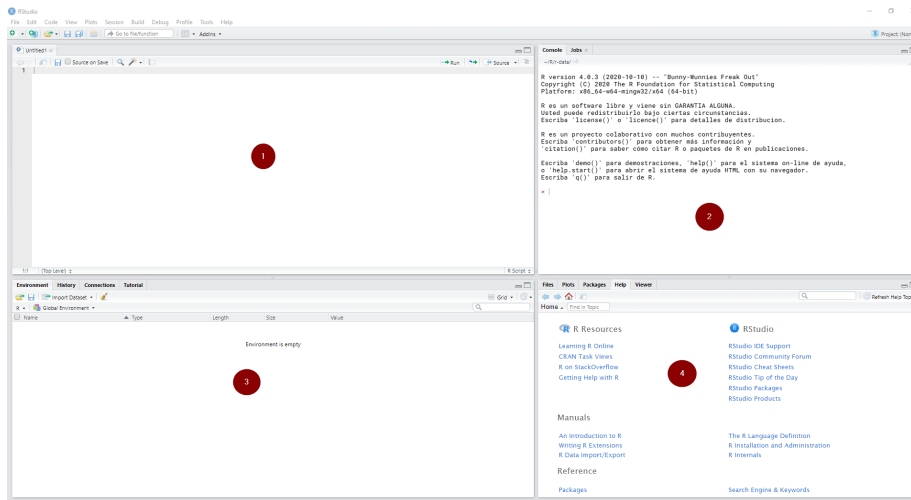
1.5.1 ¿Qué es R Studio?: una interfaz para usar R

Cuando arrancamos R Studio, se pueden ver 4 ventanas, que según el gusto del usuario, pueden estar organizadas de diversa forma. En mi caso, me gusta disponer del editor de scripts o sintaxis (*source*) en la parte superior izquierda, en la parte superior derecha la consola (*console*) y abajo de izquierda a derecha el entorno (*environment*) y a la derecha el resto de opciones (*files, plots, ...*) en la parte superior el además de la barra de opciones en la parte superior.

Puedes adoptar esta configuración desde el menú ***Tools > Global Options > Pane Layout.***



Mi escritorio ...



Ventana (1): es el editor de sintaxis: se trata del lugar donde editamos la sintaxis para posteriormente ejecutarla. Al escribir allí no sucederá nada, a no ser que se apriete algún botón para ejecutar los comandos o la tecla CTRL+ENTER.

Ventana (2): es la consola. Corresponde a lo que sería el software R en su versión básica. Allí el software ejecuta las operaciones realizadas desde el editor de sintaxis.

Ventana (4): es el “entorno de trabajo” del programa: en este lugar se muestra el conjunto de datos y los “objetos” (resultados, variables, gráficos, etc.) que se almacenan al ejecutar diferentes análisis.

Ventana (4) tiene varias sub pestañas: (i) la pestaña files permite ver el historial de archivos trabajados con el programa; (ii) la pestaña plots permite visualizar los gráficos que se generen; (iii) la pestaña packages permite ver los paquetes descargados y guardados en el disco duro así como gestionar su instalación o actualización; (iv) la ventana help permite acceder al CRAN - Comprehensive R Archive Network (siempre que se cuente con conexión a Internet), página oficial del software que ofrece diferentes recursos para el programa: manuales para el usuario, cursos on line, información general, descarga de paquetes, información de los paquetes instalados, etc. Esta última pestaña es bastante útil: empleando el motor de búsqueda se accede de manera rápida a manuales de uso de los diferentes paquetes (y sus funciones) instalados en el computador (esto no requiere conexión a Internet).⁷; (v) la ventana viewer muestra los resultados al construir reportes mediante funcionalidades tipo R Markdown que será nuestra herramienta de trabajo.

¿Dónde está nuestro trabajo? El software R funciona como un entorno temporal de trabajo, esto quiere decir que el usuario va agregando datos y objetos (conjuntos de datos con diferentes atributos) a una “hoja en blanco”. Hay que tener en cuenta que R trabaja con la memoria activa (RAM) del computador, por lo tanto cualquier análisis sólo mostrará la información resultante pero no

permanecerá como archivo posible de utilizar de modo posterior. Es decir, si los análisis no son guardados como objetos (vectores, matrices, listas u otros tipos de objetos) se deberán repetir las instrucciones para obtener otra vez el resultado.

Todas las operaciones de R - sean indicadas vía sintaxis o botones - son ejecutadas según comando computacional que es visualizado en la consola. La ejecución de comandos entrega diferentes señales respecto a su funcionamiento. Por ejemplo, mientras se está ejecutando un comando, el programa muestra un signo “Stop” en la esquina superior derecha de la consola (como se ve en la imagen). Eso indica que el programa está ocupado ejecutando una acción. Si se presiona tal símbolo, se cancelará la operación en curso.

En la sección 2, comenzamos a trabar con scscripts que nos devuelvan resultados.

Chapter 2

Primeros pasos y términos a conocer

2.1 ¿Desde dónde creo mis scripts?

Dos son las formas de trabajar con R desde R Studio. Te explicamos brevemente esas dos formas de trabajar.

2.1.1 Scripts en consola

El scripting con consola está pensado para obtener tablas u otros objetos en la consola de la interfaz de trabajo de R Studio. Para ello usaremos archivos de texto con extensión `.R` que crearemos desde la entrada *File > New File > R Script*. Estos ficheros darán como resultado salidas a la denominada consola en formato de texto.

2.1.2 Scripting con *markdown*

El scripting con *markdown* está pensado para generar páginas completas o documentos completos con texto, tablas, gráficos, etc. Se pueden utilizar paquetes como `flexdashboard` o generar salidas de tipo diapositiva utilizando las opciones de configuración que ofrece el paquete `rmarkdown`. No aseguramos una compatibilidad al 100%, pues eso sería imposible, pero gran parte de las características estarían funcionales. La idea es generar un documento HTML, de forma totalmente transparente para el usuario, que se guarda en la base de datos y se presenta como resultado del análisis. Ese HTML puede contener texto, gráficos, tablas y cualquier elemento que se te ocurra. El código R se ubica en lo

que se denomina `chunk` -que luego veremos- y se puede incluir también código R `inline` en el texto. Para esta segunda forma de trabajar usaremos también archivos de texto, pero con extensión `.Rmd` que crearemos desde la entrada *File > New File > R Markdown*. Estos ficheros darán como resultado archivos con extensiones `.html`, `.docx`, `.pdf`.

2.2 Primeros pasos

Si ya tienes experiencia con R Studio, da un vistazo, pero es posible que mucho de lo aquí indicado sea irrelevante para ti y todo ello ya sea habitual y conocido por ti. Pasa a la siguiente sección, donde comenzamos a trabajar los scripts.

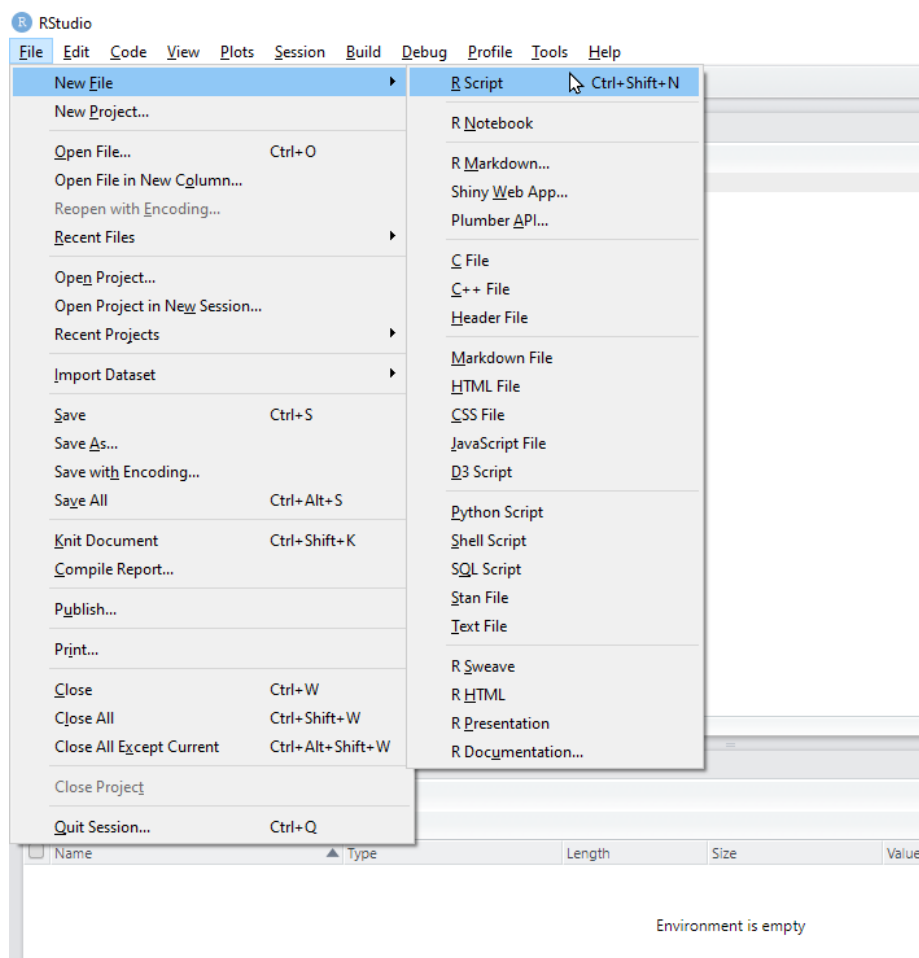
2.2.1 Carga de datos

Vamos a trabajar con ello y verás que sencillo. Para comenzar, en este enlace tienes un archivo de datos con el que haremos todos nuestros ejemplos. Esta fuente de datos está en formato SPSS (`*.sav`) y se corresponde con la tercera oleada de un estudio del CIS (Centro de Investigaciones Sociológicas de España) de 2017, el barómetro sanitario. La fuente de datos tiene 2557 registros y tiene 190 campos. Si deseas ver el cuestionario que originalmente se utilizó para recoger los datos, lo puedes descargar aquí. En él puedes ver todo lo referente a como se ha realizado la entrevista.

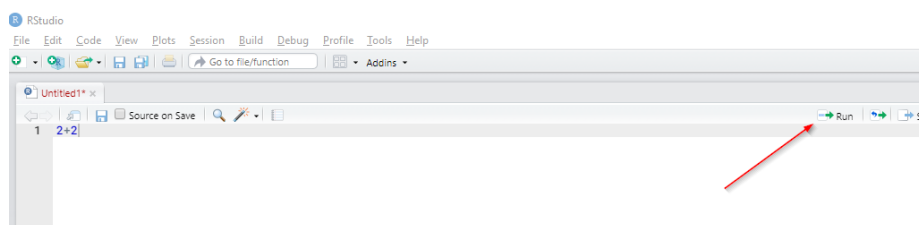
Si has seguido las instrucciones de instalación, se habrá creado una carpeta denominada R dentro de Mis Documentos (que R Studio denomina Home como podás observar en el panel de la ventana 4, opción FILES)

A partir de ahí, impera tu orden, agrega carpetas en la forma en que esté habituado y organiza tu trabajo como si de cualquier aplicación Windows se tratara. Si te gusta ser ordenado en tu trabajo, lo harás también; si no te gusta, ...

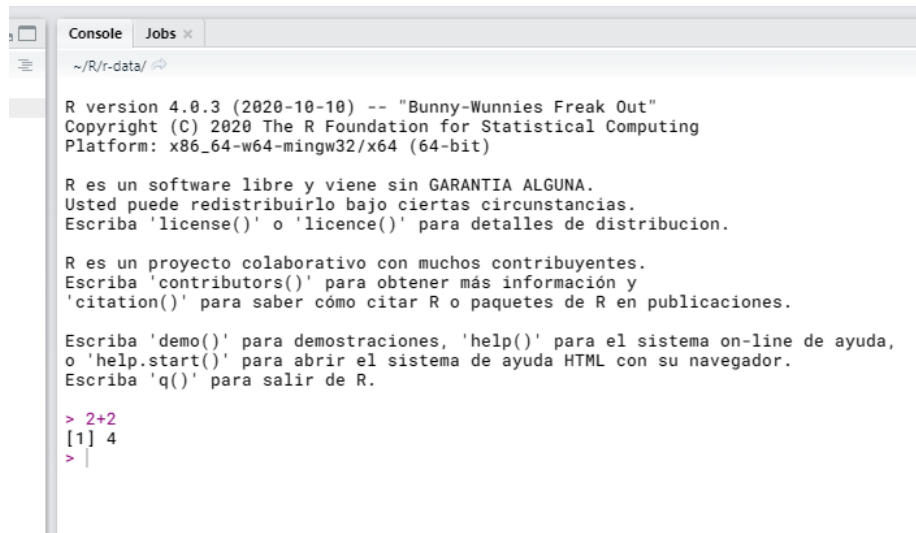
Nuestro primer paso será iniciar una sintaxis o script. Para ello desde el menú de *File > New File > R Script* abriremos un espacio para escribir. En la ventana creada como *Untitled1*, puedes escribir.



escribe $2+2$ y haz clic en *Run* teniendo el cursor sobre la línea en la que has escrito ...



y obtendrás el resultado en la consola (ventana 2)



```
Console Jobs x
~/R/r-data/ ↗

R version 4.0.3 (2020-10-10) -- "Bunny-Wunnies Freak Out"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R es un software libre y viene sin GARANTIA ALGUNA.
Usted puede redistribuirlo bajo ciertas circunstancias.
Escriba 'license()' o 'licence()' para detalles de distribucion.

R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

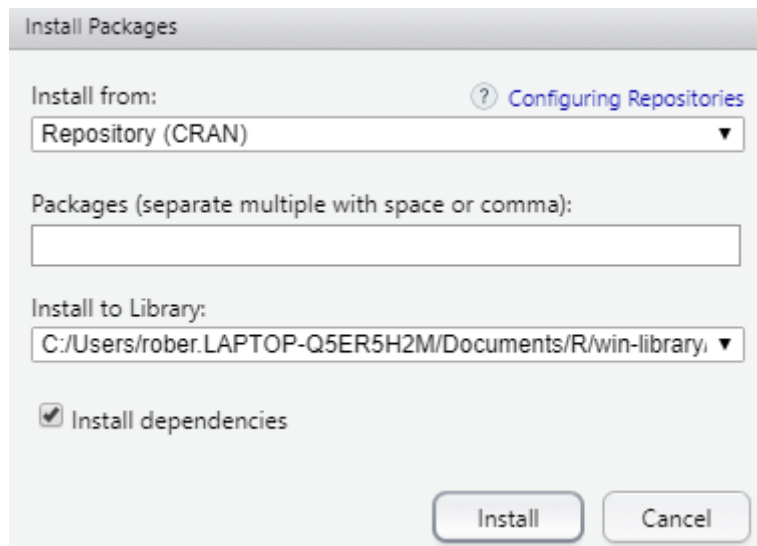
Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> 2+2
[1] 4
> |
```

Ahora podrías guardar tu script *File > Save as* , y ya tendríamos cerrado el primer paso del camino. El archivo puede abrirse, ser editado y volver a ejecutarse tantas veces quieras.

2.2.2 Instalación de paquetes

Uno de los elementos fundamentales que caracteriza a R, es que se trabaja con **paquetes** que la comunidad de desarrollo aporta para los análisis más variados. Los paquetes son la aportación de los desarrolladores que de alguna forma comparten su conocimiento con la comunidad global. Nosotros vamos a trabajar con algunos que no están en la instalación base, así que procederemos a hacer el ciclo completo de instalación (sólo una vez) y carga (tantas veces como queramos usar el paquete en una sesión). Aunque existe opción de instalar paquetes por comandos `install.packages()`, ilustramos el camino para hacerlo desde el menú. *Tools > Install packages*.



Utiliza la opción arriba indicada y siguiendo las indicaciones descritas escribiríamos:

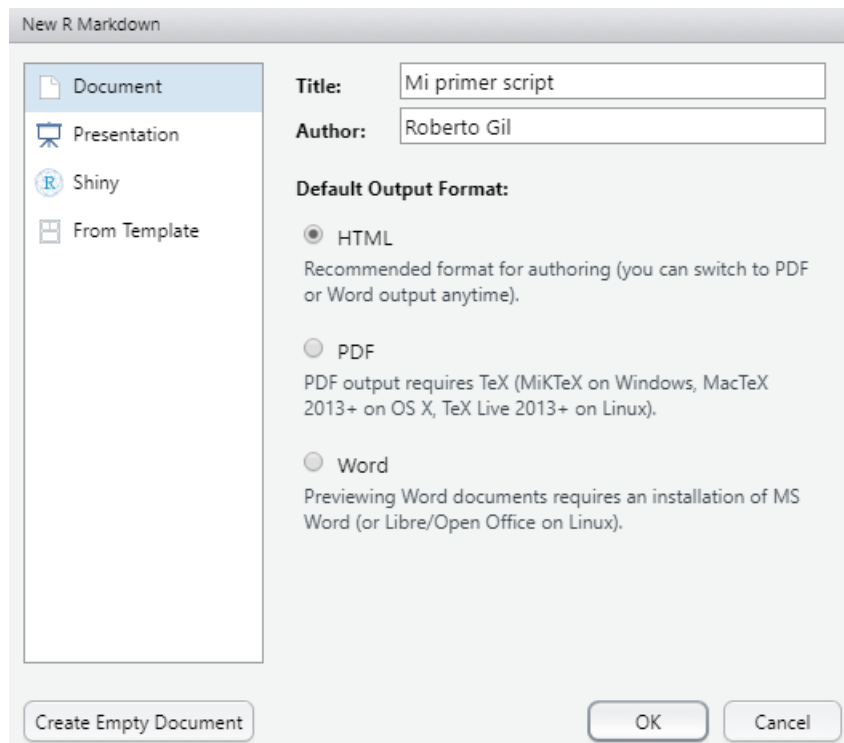
expss, highcharter, kableExtra

Cuando hagas clic en *Install*, comenzarán a instalarse esos paquetes (su descarga e instalación). R Studio te informa cuando ha finalizado. Puede verse en la consola.

2.3 Crear un fichero R Markdown

Vamos a crear un nuevo script, pero ahora ya lo vamos a hacer del modo que seguiremos trabajando durante todo el manual.

Desde el menú de R Studio **Files > New File > R Markdown**. Elige entre las opciones proporcionadas, la creación de un *documento HTML*. Dale nombre y ponte como autor. Verás lo que escribas reflejado en el archivo.



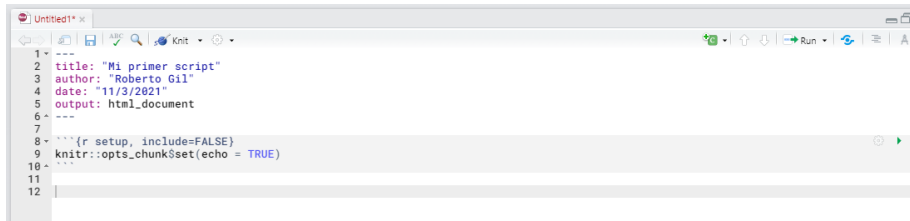
Se creará un archivo como este ...

```

1 ---
2 title: "Mi primer script"
3 author: "Roberto Gil"
4 date: "11/3/2021"
5 output: html_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 authoring HTML, PDF, and MS Word documents. For more
15 details on using R Markdown see <http://rmarkdown.rstudio.com>.
16
17 When you click the Knit button a document will be generated that includes both content as well as the output of any embedded
18 R code chunks within the document. You can embed an R code chunk like this:
19
20 ```{r cars}
21 summary(cars)
22 ```
23
24 ## Including Plots
25
26 You can also embed plots, for example:
27
28 ```{r pressure, echo=FALSE}
29 plot(pressure)
30 ```
31
32 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

```

Este es un fichero de ejemplo, borra desde donde pone `## R Markdown` hasta abajo y que quede así. Mantenemos este primer *chunk* y lo explicamos.



Aquí escribimos el script de R, que como puedes ver en línea 1 y 3 comienza y acaba con una simbología determinada. Estos son los caracteres indicadores de que todo lo que queda entre `{r echo = TRUE}` y `}` es scripting. A partir de ahora, nuestras instrucciones irán siempre entre estos símbolos de inicio y final. Nótese que se añade la instrucción `echo = TRUE`. Esta instrucción provocará que se imprima en el resultado los comandos del script, si en lugar de eso se escribe `echo=FALSE`, no se imprimiría ese código.

Nuestro código le dirá a R que secuencialmente haga...

- la carga del paquete `expss`;
- la lectura del archivo que está en esa URL (un archivo SAV - SPSS en Google Drive) y asignación a un objeto de R que lo contendrá llamado `data`;
- el cálculo de la media (`mean`)...;
- de una variable que está en el marco de datos denominado `data`;
- y que se llama PESO (`data$PESO`);
- no teniendo en cuenta los valores NA (nulos, no definidos, que no sean número: `na.rm=TRUE`)

```

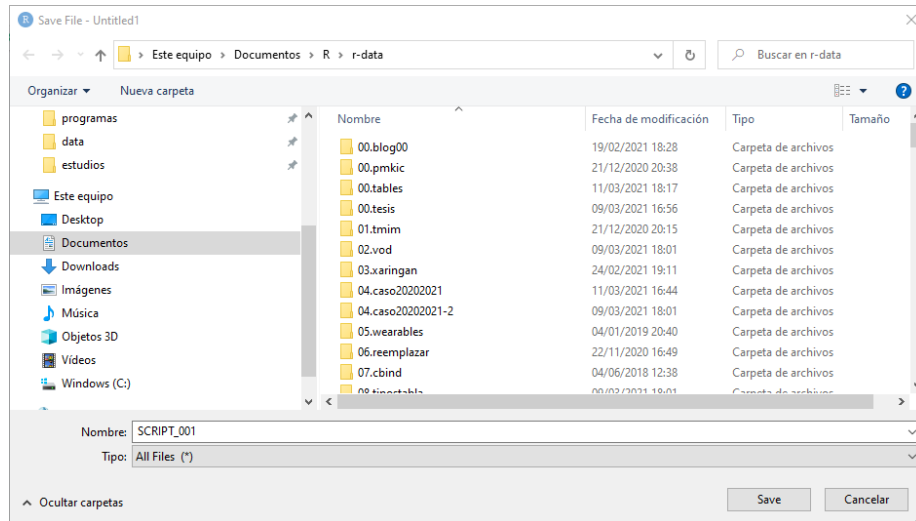
suppressMessages(setwd("~/R/r-projects/00.tables")) #esta es la carpeta donde almacené el archivo
suppressMessages(library(expss)) #cargamos el paquete
suppressMessages(data <- read_spss("data/3192.sav")) #cargamos los datos
mean(data$PESO, na.rm = TRUE)
  
```

```
## [1] 0.9999273
```

Quedará así:



Para ver el resultado, haz clic en el botón *knitr* (publicar). Como es la primera vez y no hemos guardado nuestro script, nos indica que le pongamos nombre, por ejemplo SCRIPT_001.



Y el resultado, ya puedes verlo, 0,9999273; sale en una nueva ventana que es donde se muestra el resultado. Reconocerás también aquello que se ha escrito cuando creamos el archivo y que puede editar sobre el fichero.

Mi primer script

Roberto Gil

11/3/2021

```
library(expss)
```

```
##
## Use 'expss_output_viewer()' to display tables in the RStudio Viewer.
## To return to the console output, use 'expss_output_default()'.
```

```
data <- read_expss("https://drive.google.com/uc?export=download&id=1JjevQjzGclMMyR7qz7bbvrbMyK8Fvhr")
```

```
## re-encoding from UTF-8
```

```
mean(data$PESO, na.rm=TRUE)
```

```
## [1] 0.9999273
```

Fíjate que se ha publicado el código R de programación eso lo podemos evitar si cambiamos el `echo = TRUE` por `echo = FALSE`. Prueba a hacer el cambio y haz de nuevo *knitr*.

Mi primer script

Roberto Gil

11/3/2021

```
## re-encoding from UTF-8
```

```
## [1] 0.9999273
```

Terminamos con el primer script de análisis. Este script aparece ahora en el listado de fichero como SCRIPT_001.Rmd (Rmd es la extensión de los archivos R Markdown, mientras que R es la de los scripts de consola). Del mismo modo, podrás ver ahora un archivo denominado SCRIPT_001.html que contiene tu resultado y que podrías copiar o enviar a cualquiera porque contiene todo lo necesario para que se muestre como a ti. Una de las grandes ventajas.

2.3.1 Conclusión

Así, hemos finalizado con nuestro primer *scripting*. No volveremos a ser tan explicativos en este documento acerca de como crear y editar los scripts. En la siguiente sección comenzamos con las tablas más básicas.

2.4 Básicos de R y/o proceso de datos

En el presente epígrafe, mostramos un conjunto de términos que serán habitualmente utilizados en las sucesivas secciones que se presentan en este documento. Estos son los más relevantes y los hemos separado en dos grupos. Un grupo hace referencia a términos básicos de R y otro grupo a términos básicos del manejo de tablas o del proceso de tabulación. Cada término tiene una breve reseña, y posteriormente algunos de ellos serán más tratados en sus respectivas funcionalidades.

1. **R** es un entorno y lenguaje de programación con un enfoque al análisis estadístico. Nació como una implementación de software libre del lenguaje S, adicionado con soporte para alcance estático. Se trata de uno de los lenguajes de programación más utilizados en investigación científica, siendo además muy popular en los campos de aprendizaje automático o *machine learning*, minería de datos, investigación biomédica, bioinformática y matemáticas financieras. A esto contribuye la posibilidad de cargar diferentes bibliotecas o paquetes con funcionalidades de cálculo y graficación. R es parte del sistema GNU y se distribuye bajo la licencia GNU GPL. Está disponible para los sistemas operativos Windows, Macintosh, Unix y GNU/Linux.

2. **objeto**, elemento creado desde comandos o scripts de R y que puede ser reutilizado dentro de la programación del script. En este manual se usará como sinónimo de tabla en muchos casos, pues la mayoría de los objetos que crearemos serán tablas.
3. ***dataframe***, fichero de datos, banco de datos; término con el que se conoce en R a la estructura tabular (filas y columnas) de una matriz de datos, donde las columnas son variables y las filas son registros.
4. **variable**, elemento de tipo vector que contiene los valores de una determinada observación, un valor en cada fila; debe entenderse en el contexto de la estructura tabular o *dataframe*.
5. **valores**, cada una de las diferentes celdas que componen un *dataframe*. Una variable toma un valor en cada fila y se representa en la celda.
6. **medidas**, valores de los que se pretende calcular estadísticos como la media, la desviación típica o la mediana entre otras. Suelen responder a escalas de tipo numérico (ordinal o métrico).
7. **dimensiones**, valores de los que se pretende calcular frecuencias y/o porcentajes.
8. **factores**, niveles, códigos, etiquetas de variable.
9. **NA**, es como R representa los valores nulos o ausentes.
10. **valores perdidos**, *missing values*, valores ausentes; tal como hemos indicado en el término NA, así es como R representa este tipo de valores.
11. **paquete**, conjunto de funciones de las que el usuario hace uso para obtener un resultado en R; en nuestro caso particular principalmente usaremos el paquete EXPSS de R; 1. comando, instrucciones que se integran dentro de un chunk
12. **chunk**, conjunto de comandos que se escriben entre los símbolos {r} y y que se ejecutan mostrando los resultados.

2.5 Básicos de tabulación

Antes de comenzar, el término más importante y objeto de nuestro trabajo.

Tabla

Una tabla es una matriz o cuadro que muestra la relación entre (una) dos o más variables. Cuando la tabla solo muestra la relación entre dos variables de tipo nominal u ordinal, y también se conoce como tabla de contingencia (?).

En nuestro trabajo vamos a crear objetos de tipo tabla; una tabla es una estructura tabular, igual que un *dataframe*. De hecho, con nuestro trabajo utilizando el paquete EXPSS, vamos a generar tablas que serán *dataframe* de tipo (clase) **etable**. Al ser un *dataframe*, podremos operar entre filas, columnas y celdas de forma lógica o aritmética utilizando funciones y comandos de R.

Además, dejamos este glosario de términos relacionados con las tablas en R, que utilizaremos en esta guía.

1. **título** o *caption*, texto que se publicará sobre la tabla;
2. **pie** o *footer*, texto que se publicará bajo la tabla;
3. **fila**, cada una de las líneas de información dentro de una tabla; se suele asimilar a un nivel (código) de una variable y/o a un resultados estadístico de una variable;
4. **columna**, cada una de las variables que conforman el *dataframe* de una tabla (estructura tabular); en un cuadro o tabla de contingencia suele equivaler a un nivel de la variable que originalmente se diseñó para ser usada en columnas (si por ejemplo SEXO, una columna sería hombre y otra mujer);
5. **celda**, cada una de las unidades de información del cuadro o tabla;
6. **row_label**, primera columna donde se escriben los textos de las filas y que sirven para identificar el contenido de las mismas;
7. **etiqueta de variable**, texto extra identificativo de la variable usada en filas o columnas;
8. **etiqueta de valor**, texto del código identificativo de la variable usada;
9. **estadístico**, medida calculada;
10. **frecuencia**, tipo específico de medida calculada que significa número de veces en términos absolutos;
11. **porcentaje**, tipo específico de medida calculada que significa número de veces en términos relativos;
12. **|**, símbolo denominado **pipe** que en el paquete **expss** se utilizará para separar conjuntos de texto en una celda (o columna o fila);
13. **significación**, prueba estadística de contraste.

Hasta aquí esta introducción. En la sección 3 avanzaremos en la realización de las tablas básicas o también llamadas univariantes o marginales.

Chapter 3

Tablas marginales

3.1 Una pequeña introducción

Vamos a comenzar explicando un poco qué es **expss** y su similitud nominal con **IBM SPSS**. **expss** es un paquete desarrollado por ? que calcula y muestra tablas de todo tipo, con soporte para etiquetas con estilo **SPSS** y con gran facilidad y flexibilidad para obtener cabeceras múltiples y anidadas, pesos, variables de respuesta múltiple y pruebas de significación de tabla y celda. Ofrece facilidades para una salida formateada de tablas, e incluso, aunque no es objeto de este manual la posibilidad de exportación de esas tablas a **EXCEL** con el paquete **openxlsx**. Los métodos para variables etiquetadas agregan soporte de etiquetas de valor a las funciones de R base y a algunas funciones de otros paquetes. Es un paquete destinado a ayudar a los analistas a cambiar el proceso de datos desde **EXCEL** y **SPSS** hasta R.

Aquí dejo algunos enlaces para que puedas leer acerca de este paquete y las posibilidades que te ofrece de modo combinado con R Studio:

- manual PDF de EXPSS
- material de ayuda, ejemplos
- uso de etiquetas en R

Vamos a crear nuestra primera tabla utilizando una instrucción muy básica de **expss**, que evolucionará en posteriores secciones. La que vas a ver seguidamente es la forma básica de pedir que se calcule la media de la variable PESO usando **expss**; le indicamos:

- la instrucción de cálculo *calculate*;
- el marco de datos a usar, *data*;

- y el cálculo a hacer *cro_mean* (equivalente a calcula la media *mean*) en forma de tabla.

Así pues, crea un fichero R Markdown como vimos en la sección 2, y escribe este código (o mejor *copy&paste*) y obtén el resultado ...

```
setwd("~/R/r-projects/00.tables") # esta es la carpeta donde almacené el archivo (en
library(expss) #cargamos el paquete
data <- read_spss("data/3192.sav") #cargamos los datos

## re-encoding from UTF-8

calculate(data, cro_mean(PESO)) #hacemos el cálculo
```

Table 3.1

	#Total
Ponderación	1

Verás algunos cambios respecto a la salida anterior pues no hemos indicado cuántos decimales, ni que redondeo, ni le hemos dicho que no tenga en cuenta los valores especiales o nulos... y ha respondido de forma correcta.

Ya vamos viendo que eso puede dar mucho juego, pero vamos a ir de forma ordenada y presentando poco a poco todos los tipos de tabla jugando con diferentes variables del banco de datos (¡¡¡ sí... *dataframe*!!!) que hemos cargado. Comenzaremos con la creación de tablas unidimensionales o conocidas como marginales, para luego continuar con las tablas cruzadas (sección @fig(tse04)), y entre medio, iremos incorporando medidas estadísticas.

Vamos a comenzar con un conjunto de tablas muy sencillas. En ellas representaremos los valores obtenidos del análisis de un campo extraído de nuestra fuente de datos de referencia, la tercera oleada del Barómetro Sanitario en España de 2017 del realizado y publicado por el CIS. Por ahora, trabajaremos sólo con la variable denominada P31 (sexo del entrevistado), variable medida en escala nominal, cuyas etiquetas (valores) son hombre (1) y mujer (2) y con la variable P3, escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español, medida de 1 a 10. En nuestra fuente de datos tenemos 2557 casos (entrevistas realizadas). Puedes ver estas preguntas en el cuestionario PDF que puedes bajar en la sección @fig(tse02).

Utilizaremos un script, es decir una pocas líneas de código que mostraremos en este mismo documento con un fondo gris y que lo hemos llamado *chunk*. Lo que quede fuera de ese trozo del documento (por arriba o por abajo), será como este

texto que estoy escribiendo. Este texto que además, puede ser formateado como si de un HTML se tratará, es lo que llamamos un archivo `_markdown_`, y como es de R, pues lo llamamos `_R Markdown_`. Verás que también este documento tiene títulos, que se obtienen anteponiendo el símbolo `#` desde 1 vez hasta 6 veces y que se corresponde con las etiquetas de título de HTML. Inicialmente, comentaremos las líneas del script utilizando el también el mismo símbolo, pero no al inicio de la línea sino al final. Lo que quede por detrás de él, se considera un comentario.

3.2 Frecuencias

Este conjunto de tablas sólo trabajará con el estadístico de cálculo de frecuencias. Comenzaremos con variables de respuesta simple, para luego avanzar a las variables de respuesta múltiple y al uso de medidas estadísticas básicas (suma, media, mediana, máximo, mínimo, etc.).

3.2.1 Variables de respuesta simple

3.2.1.1 Cálculo de frecuencias (estilo SPSS)

Utilizaremos en estos ejemplos de forma inicial un campo del marco de datos, `P31`, de respuesta simple. La primera tabla que haremos responde a un recuento de frecuencias, y es muy usada para el análisis univariante de una campo. Este comando muestra una tabla básica utilizando la función `fre()` que copia la salida del SPSS. Nótese que la columna de porcentaje válido y porcentaje es igual ante la inexistencia de NA (valores perdidos).

```
tab <- fre(data$P31)
as.datatable_widget(tab)
```

Sexo de la persona entrevistada	Count	Valid percent	Percent	Responses, %	Cumulative responses, %
Hombre	1256	49.1	49.1	49.1	49.1
Mujer	1301	50.9	50.9	50.9	100
#Total	2557	100	100	100	
<NA>	0		0		

Figure 3.1: Frecuencias marginales de P31, estilo SPSS

Alternativamente se puede presentar la forma que trabajaremos a lo largo de este curso, esta forma es la denominada script encadenado, donde definimos el marco de datos al inicio, y encadenamos instrucciones con el símbolo `%>%` que irían línea a línea sucesivamente para una mejor lectura y comprensión del texto escrito; podrían perfectamente ir en una línea. Nótese que la tabla sale igual con las dos formas, pero mientras que en el primer caso se usa la nomenclatura estándar de R, y el campo se llama `data$P31`, es decir nombre del marco de datos en R (`data`) el símbolo del `$` que separa y nombre del campo en el marco de datos `P31` en la segunda al definir de inicio que se utilizará `data` ya se usa el nombre `P31` directamente, aunque debamos dar la orden de cálculo con el comando `calculate()`.

```
as.datatable_widget(data %>%
  calculate(fre(P31)))
```

Sexo de la persona entrevistada	Count	Valid percent	Percent	Responses, %	Cumulative responses, %
Hombre	1256	49.1	49.1	49.1	49.1
Mujer	1301	50.9	50.9	50.9	100
#Total	2557	100	100	100	
<NA>	0		0		

Figure 3.2: Frecuencias marginalesde P31 en tabla

Veamos ahora cómo solicitaremos tablas de frecuencias, porcentajes y estadísticos simples con R.

3.2.1.2 Tablas de frecuencias (absolutos)

La segunda tabla que vamos a hacer, ya responde a la típica presentación de una tabla de contingencia, sólo que en este casos vamos a mostrar sólo un campo y por tanto no va a haber cruce de variables. En el paquete `expss`, para construir un cuadro deberemos indicar al menos:

- un marco de datos (*dataframe* en nomenclatura R)
- referenciar la variable sobre la que se deben calcular el estadístico seleccionado (frecuencia -casos-, media, mediana, máximo, mínimo...)
- una orden de impresión de tabla

Estos elementos básicos pueden completarse con campos de columnas, campos de filas, pruebas de significación, etc. Iremos desarrollando estos conceptos a lo largo de este documento. ¡Vamos a por el cuadro!

La que ahora entregamos, es la estructura básica de un script de R con el paquete `expss`. A lo largo del documento veremos cómo ir introduciendo mínimas variaciones a esta estructura que te permitirán descubrir un sinnúmero de posibilidades que ofrece este paquete de R. Por ejemplo, podemos modificar la etiqueta de TOTAL o indicar donde debe situarse la fila que contiene el cálculo TOTAL. Todas estas posibilidades las puedes conocer en la documentación original del *package*, aunque en este manual trataremos de ir desgranado las

más relevantes para nuestro objetivo. Inicialmente iremos añadiendo tras el operador `%>%` comentarios precedidos por el símbolo `#`. Estos comentarios irán desapareciendo a medida que avancemos en el manual, y sólo se recurrirá a ellos cuando se aporte alguna nueva funcionalidad.

Para este primer script, indicaremos que usamos la fuente de datos (*dataframe*) ya cargado en el análisis. En R Studio, el *dataframe* tendrá el nombre que le hayas indicado en la carga -en nuestro caso *data*-. Redactamos pues nuestro script, donde identificamos el *dataframe*, el campo **P31** del cual vamos a calcular el número de casos:

```
as.datatable_widget(data %>%
  tab_cells(P31) %>%
  tab_stat_cases() %>%
  tab_pivot())
```

		#Total
Sexo de la persona entrevistada	Hombre	1256
	Mujer	1301
#Total cases		2557

Figure 3.3: Frecuencias de P31

Realicemos ahora una pequeña pero importante variación en el cálculo del estadístico casos -frecuencias- y utilicemos la posibilidad de ubicar donde queramos el total de casos, así como su etiqueta. Ello lo hacemos con `total_row_position = "above", label = "Casos"` aplicado a la función `tab_stat_cases()`.

```
as.datatable_widget(data %>%
  tab_cells(P31) %>%
  tab_stat_cases(total_row_position = "above", label = "Casos") %>%
  tab_pivot())
```

			#Total
Sexo de la persona entrevistada	#Total cases	Casos	2557
	Hombre	Casos	1256
	Mujer	Casos	1301

Figure 3.4: Frecuencias de P31, moviendo el Total

3.2.1.3 Tablas de frecuencias relativas

Si en lugar de obtener casos (valores absolutos) queremos sacar valores porcentuales, el cambio es mínimo. Usaremos el comando `tab_stat_cpct()` para indicarlo.

```
as.datatable_widget(data %>%  
  tab_cells(P31) %>%  
  tab_stat_cpct(total_row_position = "above", label = "% casos") %>%  
  tab_pivot())
```

			#Total
Sexo de la persona entrevistada	#Total casos	% casos	2557
	Hombre	% casos	49.1
	Mujer	% casos	50.9

Figure 3.5: Porcentajes de P31

3.2.1.4 Tablas de absolutos y relativos (juntos)

Cuando deseamos hacer combinaciones de frecuencias y porcentajes, la filosofía de trabajo es muy parecida. En nuestro caso vamos a hacer algo muy típico. Aunque creo que resulta más sencillo leer cada estadístico en su tabla, hay ocasiones en las que la comparativa es muy necesaria y por tanto es necesario unir los estadísticos en la misma tabla. Nótese la diferencia con el siguiente cuadro...

```
as.datatable_widget(data %>%
  tab_cells(P31) %>%
  tab_stat_cases(total_row_position = "above", label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_pivot(stat_position = "inside_columns"))
```


		#Total	
		Casos	% casos
Sexo de la persona entrevistada	#Total cases	2557	2557
	Hombre	1256	49.1
	Mujer	1301	50.9

Figure 3.6: Frecuencias y porcentajes de P31

Nótese el efecto introducido por el modificador de posición del cálculo. También ...

```
as.datatable_widget(data %>%
  tab_cells(P31) %>%
  tab_stat_cases(total_row_position = "above", label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_pivot(stat_position = "outside_rows"))
```

			#Total
Sexo de la persona entrevistada	#Total cases	Casos	2557
	Hombre	Casos	1256
	Mujer	Casos	1301
	Hombre	% casos	49.1
	Mujer	% casos	50.9
	#Total cases	% casos	2557

Figure 3.7: Tablas con propiedades diferentes a estándar

O también ...

```
as.datatable_widget(data %>%
  tab_cells(P31) %>%
  tab_stat_cases(total_row_position = "below", label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_pivot(stat_position = "inside_columns"))
```

		#Total	
		Casos	% casos
Sexo de la persona entrevistada	Hombre	1256	49.1
	Mujer	1301	50.9
#Total cases		2557	2557

Figure 3.8: Propiedades diferentes al estándar

3.2.2 Variable de respuesta múltiple

Vamos a trabajar ahora con variables multi respuesta. Para trabajar con múltiples, debemos conocer en qué forma nos llegan en nuestro *input*. Por ejemplo, SPSS divide la variable múltiple en tantas variables simples (o dicotómicas binarias) como requiera para poder representar la multi respuesta. Por ejemplo, si tenemos una variable múltiple denominada P01, y el máximo número de respuestas (menciones) en el banco de datos es 3, al crear el `_dataframe_` se crean las variables P01_1, P01_2 y P01_3; es con estas variables con las que trabajamos. Cada una de estas variables puede tomar cualquiera de los valores codificados.

Para `expss`, la forma de indicar que un conjunto de campos forman una multi respuesta es muy simple anteponer `mrset_f()` al nombre del campo que vamos a usar. Debemos tener la precaución de que no haya variables en el banco de datos que comiencen por la misma raíz. Así, el campo de ejemplo sería `mrset_f(P01_)` y con eso procesaría las tres variables de forma conjunta. Alternativamente, podríamos usar también:

- `mrset(P01_1 %to% P01_3)` o también,
- `mrset(P01_1,P01_2,P01_3)`

Cualquiera de ellas sería también válida, pero nótese que en estas últimas listadas, es necesario saber donde empieza y acaba la múltiple y esto puede variar sobretodo si creamos los script antes de acabar el campo. Al acabar el campo, pudiera haber algún nuevo caso que tuviera más menciones que 3 y por tanto existirían también `_4`, `_5` o, `_n`.

Como hemos indicado, no olvides que existe otra forma de trabajar las múltiples, utilizando variables dicotómicas o binarias (así es como están en nuestro banco de datos del CIS). En este caso, serviría todo lo afirmado anteriormente, pero en lugar de `mrset_f()`, usaríamos `mdset_f()`.

3.2.2.1 Tablas de frecuencias absolutas

Usaremos el campo P18C para procesar su información, que se localiza en el banco de datos desde P18C01 hasta P18C08.

```
as.datatable_widget(data %>%
  tab_cells(mdset_f(P18C)) %>%
  tab_stat_cases(total_row_position = "above", label = "Casos") %>%
  tab_pivot(stat_position = "inside_columns"))
```

	#Total
	Casos
#Total cases	198
La tarjeta sanitaria no funcionaba	33
El ordenador de la farmacia no funcionaba	23
Estaba fuera de plazo (era demasiado pronto o demasiado tarde)	75
No aparecían los medicamentos recetados	57
No pudo retirarlos en una comunidad autónoma distinta a la suya	17
Otro tipo de problema	48
No recuerda	
N.C.	3

Figure 3.9: Frecuencias de P18

3.2.2.2 Tablas de frecuencias relativas

También se pueden, como es obvio, obtener porcentajes en las tablas marginales múltiples. A diferencia de cuando la variables es simple que todos los porcentajes suman 100, en las variables múltiples cada alternativa tiene un rango de 0 a 100, desde no ser elegida una opción en ningún registro del `_dataframe_`, hasta ser elegida por todos los registros. Usaremos nuevamente el campo P18C para procesar su información, que se localiza en el banco de datos desde P18C01 hasta P18C08.

```
as.datatable_widget(data %>%
  tab_cells(mdset_f(P18C)) %>%
  tab_stat_cpct(total_row_position = "above", label = "% casos") %>%
  tab_pivot(stat_position = "inside_columns"))
```

	#Total
	% casos
#Total cases	198
La tarjeta sanitaria no funcionaba	16.7
El ordenador de la farmacia no funcionaba	11.6
Estaba fuera de plazo (era demasiado pronto o demasiado tarde)	37.9
No aparecían los medicamentos recetados	28.8
No pudo retirarlos en una comunidad autónoma distinta a la suya	8.6
Otro tipo de problema	24.2
No recuerda	
N.C.	1.5

Figure 3.10: Frecuencias de P18

Pero vamos a introducir una nueva variación. En una múltiple, también pueden calcularse los resultados en lo que se llama **base respuestas**, donde sí suman 100% los porcentajes nuevamente, pero recuerda que el porcentaje hace referencia a las respuestas, no a los individuos. En este caso el script modifica el estadístico solicitado.

```
as.datatable_widget(data %>%
  tab_cells(mdset_f(P18C)) %>%
  tab_stat_cpct_responses(total_row_position = "above",
    label = "% casos") %>%
  tab_pivot(stat_position = "inside_columns"))
```

	#Total
	% casos
#Total responses	256
La tarjeta sanitaria no funcionaba	12.9
El ordenador de la farmacia no funcionaba	9
Estaba fuera de plazo (era demasiado pronto o demasiado tarde)	29.3
No aparecían los medicamentos recetados	22.3
No pudo retirarlos en una comunidad autónoma distinta a la suya	6.6
Otro tipo de problema	18.8
No recuerda	
N.C.	1.2

Figure 3.11: Frecuencias de P18

3.2.3 Tablas combinadas

Con las múltiples también funciona el posicionamiento del estadístico casos - frecuencias- cuando combinamos los mismos (frecuencia y porcentaje) y podemos realizar las mismas variantes que antes.

Ubicar los cálculos dentro de las columnas ...

```
as.datatable_widget(data %>%
  tab_cells(mdset_f(P18C)) %>% # o tab_cells(mdset(P18C01 %to% P18C08))
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_stat_cpct_responses(label = "% respuestas") %>%
  tab_pivot(stat_position = "inside_columns"))
```

	#Total		
	Casos	% casos	% respuestas
La tarjeta sanitaria no funcionaba	33	16.7	12.9
El ordenador de la farmacia no funcionaba	23	11.6	9
Estaba fuera de plazo (era demasiado pronto o demasiado tarde)	75	37.9	29.3
No aparecían los medicamentos recetados	57	28.8	22.3
No pudo retirarlos en una comunidad autónoma distinta a la suya	17	8.6	6.6
Otro tipo de problema	48	24.2	18.8
No recuerda			
N.C.	3	1.5	1.2
#Total responses			256
#Total cases	198	198	

Figure 3.12: Frecuencias y porcentajes de P18 (1)

Préstese atención a las dos líneas de #Total, dado que las bases son diferentes (número de individuos y número de respuestas).

Podemos ubicar los cálculos dentro de las filas ...

```
as.datatable_widget(data %>%
  tab_cells(mdset_f(P18C)) %>% # o tab_cells(mdset(P18C01 %to% P18C08))
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_stat_cpct_responses(label = "% respuestas") %>%
  tab_pivot(stat_position = "inside_rows"))
```

		#Total
La tarjeta sanitaria no funcionaba	Casos	33
	% casos	16.7
	% respuestas	12.9
El ordenador de la farmacia no funcionaba	Casos	23
	% casos	11.6
	% respuestas	9
Estaba fuera de plazo (era demasiado pronto o demasiado tarde)	Casos	75
	% casos	37.9
	% respuestas	29.3
No aparecían los medicamentos recetados	Casos	57
	% casos	28.8
	% respuestas	22.3
No pudo retirarlos en una comunidad autónoma distinta a la suya	Casos	17
	% casos	8.6
	% respuestas	6.6
Otro tipo de problema	Casos	48
	% casos	24.2
	% respuestas	18.8
No recuerda	Casos	
	% casos	
	% respuestas	
N.C.	Casos	3
	% casos	1.5
	% respuestas	1.2
#Total cases	Casos	198
	% casos	198
#Total responses	% respuestas	256

Figure 3.13: Frecuencias y porcentajes de P18 (2)

Podemos ubicar los cálculos fuera de las columnas (igual a la anterior `inside...` porque no hay campo de columna) ...

```
as.datatable_widget(data %>%
  tab_cells(mdset_f(P18C)) %>% # o tab_cells(mdset(P18C01 %to% P18C08))
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_stat_cpct_responses(label = "% respuestas") %>%
  tab_pivot(stat_position = "outside_columns"))
```


	#Total		
	Casos	% casos	% respuestas
La tarjeta sanitaria no funcionaba	33	16.7	12.9
El ordenador de la farmacia no funcionaba	23	11.6	9
Estaba fuera de plazo (era demasiado pronto o demasiado tarde)	75	37.9	29.3
No aparecían los medicamentos recetados	57	28.8	22.3
No pudo retirarlos en una comunidad autónoma distinta a la suya	17	8.6	6.6
Otro tipo de problema	48	24.2	18.8
No recuerda			
N.C.	3	1.5	1.2
#Total responses			256
#Total cases	198	198	

Figure 3.14: Frecuencias y porcentajes de P18 (3)

Podemos ubicar los cálculos fuera de las filas ... nótese que la agrupación es diferente a la anterior con `inside_rows`

```
as.datatable_widget(data %>%
  tab_cells(mdset_f(P18C)) %>% # o tab_cells(mdset(P18C01 %to% P18C08))
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_stat_cpct_responses(label = "% respuestas") %>%
  tab_pivot(stat_position = "outside_rows"))
```

		#Total
La tarjeta sanitaria no funcionaba	Casos	33
El ordenador de la farmacia no funcionaba	Casos	23
Estaba fuera de plazo (era demasiado pronto o demasiado tarde)	Casos	75
No aparecían los medicamentos recetados	Casos	57
No pudo retirarlos en una comunidad autónoma distinta a la suya	Casos	17
Otro tipo de problema	Casos	48
No recuerda	Casos	
N.C.	Casos	3
#Total cases	Casos	198
La tarjeta sanitaria no funcionaba	% casos	16.7
El ordenador de la farmacia no funcionaba	% casos	11.6
Estaba fuera de plazo (era demasiado pronto o demasiado tarde)	% casos	37.9
No aparecían los medicamentos recetados	% casos	28.8
No pudo retirarlos en una comunidad autónoma distinta a la suya	% casos	8.6
Otro tipo de problema	% casos	24.2
No recuerda	% casos	
N.C.	% casos	1.5
#Total cases	% casos	198
La tarjeta sanitaria no funcionaba	% respuestas	12.9
El ordenador de la farmacia no funcionaba	% respuestas	9
Estaba fuera de plazo (era demasiado pronto o demasiado tarde)	% respuestas	29.3
No aparecían los medicamentos recetados	% respuestas	22.3
No pudo retirarlos en una comunidad autónoma distinta a la suya	% respuestas	6.6
Otro tipo de problema	% respuestas	18.8
No recuerda	% respuestas	
N.C.	% respuestas	1.2
#Total responses	% respuestas	256

Figure 3.15: Frecuencias y porcentajes de P18 (4)

3.3 Estadísticos

Hasta ahora hemos trabajado sólo con casos, pero ya hemos anticipado que al igual que con los recuentos de casos o frecuencias se puede trabajar con otros estadísticos como la suma, máximo, mínimo, media, mediana, error estándar y desviación típica. Vamos a ir viendo cómo se desarrollan estos cuadros.

3.3.1 Estadísticos básicos

Recordemos que hasta ahora no hemos cruzado la información, solo estamos trabajando con lo que se denomina medidas marginales. Nuestro primer ejemplo es un caso típico, donde queremos obtener la media (`tab_stat_mean`), la desviación típica (`tab_stat_sd()`) y la base de cálculo, es decir el número de casos con valor (`tab_stat_valid_n()`) para el cálculo.

Así, siguiendo la misma estructura de las tablas anteriores, redactamos el siguiente script:

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_stat_mean() %>%
  tab_stat_sd() %>%
  tab_stat_valid_n() %>%
  tab_pivot())
```

	#Total	
Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español	Mean	7.3
	Std. dev.	7.2
	Valid N	2557

Figure 3.16: Estadísticos marginales de P3 (1)

No, no tienes por qué ver los nombres de los estadísticos en lengua inglesa. También aquí podemos jugar con la etiqueta (label).

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_stat_mean(label = "media") %>%
  tab_stat_sd(label = "desviación") %>%
  tab_stat_valid_n(label = "casos") %>%
  tab_pivot())
```

	#Total
Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español	
media	7.3
desviación	7.2
casos	2557

Figure 3.17: Estadísticos marginales de P3 (2)

Hagamos una nueva tabla con una pequeña variación, ahora vamos a poner los estadísticos en columnas.

```
as.datatable_widget(data %>%  
  tab_cells(P3) %>%  
  tab_stat_mean(label = "media") %>%  
  tab_stat_sd(label = "desviación") %>%  
  tab_stat_valid_n(label = "casos") %>%  
  tab_pivot(stat_position = "inside_columns"))
```

	#Total		
	media	desviación	casos
Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español	7.3	7.2	2557

Figure 3.18: Estadísticos marginales de P3 (3)

`expss` tiene además la posibilidad de obtener estos tres cálculos, bastante habituales por cierto, con un solo comando: `tab_stat_mean_sd_n()` pudiendo añadir además etiquetas separadas.

```
as.datatable_widget(data %>%  
  tab_cells(P3) %>%  
  tab_stat_mean_sd_n(labels = c("media", "desviación",  
    "casos")) %>%  
  tab_pivot())
```

	#Total
Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español	
media	7.3
desviación	7.2
casos	2557

Figure 3.19: Estadísticos marginales de P3 (4)

3.3.2 Otros estadísticos

Además de los estadísticos más básicos, otros que podemos añadir son el máximo, el mínimo, la mediana, el error estándar y la suma. Los unimos todos.

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_stat_max(label = "Máximo") %>%
  tab_stat_min(label = "Mínimo") %>%
  tab_stat_median(label = "Mediana") %>%
  tab_stat_se(label = "Error estándar") %>%
  tab_stat_sum(label = "Suma") %>%
  tab_pivot())
```

	# Total
Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español	
Media	7.3
Desviación	7.2
Máximo	99
Mínimo	1
Mediana	7
Error estándar	0.1
Suma	18789

Figure 3.20: Estadísticos marginales de P3 (5)

Nótese que no se han definido ni filas, ni columnas. Es el modificador de la posición de los estadísticos (`stat_position`) el que habilita la posición en una fila.

Del mismo modo, estos estadísticos pueden ubicarse en las columnas.

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_stat_max(label = "Máximo") %>%
  tab_stat_min(label = "Mínimo") %>%
  tab_stat_median(label = "Mediana") %>%
  tab_stat_se(label = "Error estándar") %>%
  tab_stat_sum(label = "Suma") %>%
  tab_stat_cases(label = "casos") %>%
  tab_pivot(stat_position = "inside_rows"))
```

		#Total
Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español	Media	7.3
	Desviación	7.2
	Máximo	99
	Mínimo	1
	Mediana	7
	Error estándar	0.1
	Suma	18789
	1 Muy insatisfecho/a	casos 40
	2	casos 35
	3	casos 75
	4	casos 111
	5	casos 295
	6	casos 397
	7	casos 584
	8	casos 623
	9	casos 210
	10 Muy satisfecho/a	casos 172
	N.S.	casos 14
	N.C.	casos 1
	#Total cases	casos 2557

Figure 3.21: Estadísticos marginales de P3 (6)

Hagamos finalmente una leve variación. Nótese que al utilizar "|" en la etiqueta del estadístico casos, hemos eliminado la columna intermedia y aparece todo como más compacto. Este será un recurso que utilizaremos en muchas ocasiones.

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_stat_max(label = "Máximo") %>%
  tab_stat_min(label = "Mínimo") %>%
  tab_stat_median(label = "Mediana") %>%
  tab_stat_se(label = "Error estándar") %>%
  tab_stat_sum(label = "Suma") %>%
  tab_stat_cases(label = "|") %>%
  tab_pivot(stat_position = "inside_rows"))
```


	#Total
Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español	
Media	7.3
Desviación	7.2
Máximo	99
Mínimo	1
Mediana	7
Error estándar	0.1
Suma	18789
1 Muy insatisfecho/a	40
2	35
3	75
4	111
5	295
6	397
7	584
8	623
9	210
10 Muy satisfecho/a	172
N.S.	14
N.C.	1
#Total cases	2557

Figure 3.22: Estadísticos marginales de P3 (7)

3.4 Conclusión

Creo que esta primera muestra de cómo procesar nuestra tabla de una única variable, es más que suficiente para colmar las expectativas más exigentes. Para aquellos que conozcan un poco más el funcionamiento de R, indicar que cada una de estas tablas, se puede almacenar como objeto sobre el que se puede trabajar. Este objeto es del tipo `etable` pero en el fondo es un objeto de tipo `_dataframe_` que por tanto puedes ser trabajado con comandos R estándar. Es de esta posibilidad de ser un `dataframe` de donde deriva su capacidad de integración con otros paquetes como por ejemplo `highcharter` ? que será uno de nuestros paquetes de referencia para gráficos. Para una presentación completa, véase la sección ?? para una presentación de gráficos a partir de `_dataframe_` o de tablas cruzadas - `_crosstab_` -.

Hasta llegar ese momento, ahora en la siguiente sección 4 analizamos las tablas cruzadas.

Chapter 4

Tablas cruzadas

A diferencia de lo visto en la sección 3, en este epígrafe analizaremos como obtener cuadros resumen en los que existen variables en la cabecera, que determinan grupos o perfiles de análisis y existen variables en las filas, de las cuáles queremos conocer cómo se distribuyen sus alternativas de respuesta entre los diferentes perfiles o grupos que determinan las variables de columna. Al igual que sucedió con las tablas marginales, mostraremos poco a poco como trabajar con variables de respuesta simple, múltiple o con medidas estadísticas. Vamos a utilizar otros campos que se localizan en la base de datos del CIS (P3, P21A01, P21A02, P21A03, P31, P33). Aquí su resumen... -nótese que se ha incluido la función `suppressMessages()` para no publicar los mensajes de carga del paquete y datos-.

```
suppressMessages(setwd("~/R/r-projects/00.tables")) #esta es la carpeta donde almacené el archivo
suppressMessages(library(expss)) #cargamos el paquete
suppressMessages(data <- read_spss("data/3192.sav")) #cargamos los datos
data$P3
```

```
## LABEL: Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español
## VALUES:
## 6, 7, 8, 8, 7, 5, 7, 7, 8, 7, 7, 6, 6, 7, 7, 6, 5, 8, 7, 7, 8, 10, 8, 6, 6, 9, 8, 8, 7, 6, 9,
## VALUE LABELS:
## 1 1 Muy insatisfecho/a
## 2 2
## 3 3
## 4 4
## 5 5
## 6 6
## 7 7
## 8 8
```

```
## 9 9
## 10 10 Muy satisfecho/a
## 98 N.S.
## 99 N.C.
```

```
data$P21A01
```

```
## LABEL: Medicamentos que recetan por adelantado (para que no falten)
## VALUES:
## 0, 0, 0, 0, 0, 0, NA, 0, 0, 0, 0, 0, 0, 0, 0, NA, 0, NA, 0, 0, NA, 0, 0, 0, 0, 0
## VALUE LABELS:
## 0 N.P.
## 1 Menciona
```

```
data$P21A02
```

```
## LABEL: Envases que han quedado sin usar porque cambiaron el tratamiento
## VALUES:
## 0, 0, 0, 0, 0, 0, NA, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, NA, 0, 0, 1, 0, 0, 0, 0, 0
## VALUE LABELS:
## 0 N.P.
## 1 Menciona
```

```
data$P21A03
```

```
## LABEL: Medicamentos que decidió no tomar
## VALUES:
## 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, NA, 0, 0, 0, 0, 0
## VALUE LABELS:
## 0 N.P.
## 1 Menciona
```

```
data$P31
```

```
## LABEL: Sexo de la persona entrevistada
## VALUES:
## 2, 1, 2, 1, 2, 2, 1, 1, 2, 2, 1, 1, 2, 1, 2, 2, 2, 1, 2, 2, 2, 1, 2, 1, 1, 2, 1, 1
## VALUE LABELS:
## 1 Hombre
## 2 Mujer
```

```
data$P33
```

```
## LABEL: Estado civil de la persona entrevistada
## VALUES:
## 1, 1, 1, 1, 3, 2, 2, 2, 1, 9, 1, 2, 1, 1, 3, 2, 3, 1, 1, 1, 1, 1, 1, 2, 2, 3, 2, 1, 1, 2, 2, 2
## VALUE LABELS:
## 1 Casado/a
## 2 Soltero/a
## 3 Viudo/a
## 4 Separado/a
## 5 Divorciado/a
## 9 N.C.
```

4.1 Básica con variables simples

Las tablas que vamos a hacer a continuación, siempre son tablas en las que intervienen al menos dos variables. Una de las variables irá a columnas y la otra variable irá a filas. De ellas se calcularán las frecuencias absolutas o relativas y/o los estadísticos. Vamos a empezar sólo con el estadístico frecuencias, y posteriormente ya pasaremos a estadísticos como la media, suma, etc...

4.1.1 De frecuencias, variable simple y sólo absolutos

Usaremos la variable P31 para las columnas, P33 en las celdas (de la que se hará el cálculo) y el estadístico a usar será el n^o de casos.

```
as.datatable_widget(data %>%
  tab_cells(P33) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cases(total_row_position = "above", total_label = "Total") %>%
  tab_pivot())
```

		#Total	Sexo de la persona entrevistada		
			Hombre	Mujer	
Estado civil de la persona entrevistada	#Total	2557	1256		1301
	Casado/a	1388	677		711
	Soltero/a	817	455		362
	Viado/a	190	41		149
	Separado/a	57	24		33
	Divorciado/a	97	55		42
	N.C.	8	4		4

Figure 4.1: Cruce de estado civil por sexo

4.1.2 De frecuencias, variable simple, con porcentajes de columna

Al igual que hicimos con las tablas marginales vamos a repetir esta tabla, pero en porcentaje de columna (vertical).

```
as.datatable_widget(data %>%
  tab_cells(P33) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cpct(total_row_position = "above", total_label = "Total") %>% # aquí seña
  tab_pivot())
```

		#Total	Sexo de la persona entrevistada		
			Hombre	Mujer	
Estado civil de la persona entrevistada	#Total	2557	1256	1301	
	Casado/a	54.3	53.9	54.7	
	Soltero/a	32	36.2	27.8	
	Viado/a	7.4	3.3	11.5	
	Separado/a	2.2	1.9	2.5	
	Divorciado/a	3.8	4.4	3.2	
	N.C.	0.3	0.3	0.3	

Figure 4.2: Cruce de estado civil por sexo, porcentaje de columna

4.1.3 De frecuencias, variable simple, con porcentajes de fila

Al igual que hicimos con las tablas marginales vamos a repetir esta tabla, pero en porcentaje de fila (horizontal).

```
as.datatable_widget(data %>%
  tab_cells(P33) %>%
  tab_cols(total(), P31) %>%
  tab_stat_rpct(total_row_position = "above", total_label = "Total") %>% # aquí señalo los porcentajes de fila
  tab_pivot())
```

		#Total	Sexo de la persona entrevistada		
			Hombre	Mujer	
Estado civil de la persona entrevistada	#Total	2557	1256		1301
	Casado/a	100	48.8		51.2
	Soltero/a	100	55.7		44.3
	Viado/a	100	21.6		78.4
	Separado/a	100	42.1		57.9
	Divorciado/a	100	56.7		43.3
	N.C.	100	50		50

Figure 4.3: Cruce de estado civil por sexo, porcentaje de fila

4.1.4 De frecuencias, variable simple, con porcentajes total muestra

Al igual que hicimos con las tablas marginales vamos a repetir esta tabla, pero en porcentaje sobre el total de la muestra.

```
as.datatable_widget(data %>%
  tab_cells(P33) %>%
  tab_cols(total(), P31) %>%
  tab_stat_tpct(total_row_position = "above", total_label = "Total") %>% # aquí seña
  tab_pivot())
```


		#Total	Sexo de la persona entrevistada		
			Hombre	Mujer	
Estado civil de la persona entrevistada	#Total	2557	1256	1301	
	Casado/a	54.3	26.5	27.8	
	Soltero/a	32	17.8	14.2	
	Viado/a	7.4	1.6	5.8	
	Separado/a	2.2	0.9	1.3	
	Divorciado/a	3.8	2.2	1.6	
	N.C.	0.3	0.2	0.2	

Figure 4.4: Cruce de estado civil por sexo, porcentaje sobre el total de la muestra

4.1.5 Combinaciones de los anteriores

Vamos ahora a hacer combinaciones entre ellos. Advierto que cada vez se dificulta más la tabla en su lectura. Como dije inicialmente, me decanto más por tablas sencillas y con un sólo dato.

```
as.datatable_widget(data %>%
  tab_cells(P33) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_pivot(stat_position = "inside_columns"))
```

		#Total		Sexo de la persona entrevistada			
		Casos	% casos	Hombre		Mujer	
				Casos	% casos	Casos	% casos
Estado civil de la persona entrevistada	Casado/a	1388	54.3	677	53.9	711	54.7
	Soltero/a	817	32	455	36.2	362	27.8
	Viado/a	190	7.4	41	3.3	149	11.5
	Separado/a	57	2.2	24	1.9	33	2.5
	Divorciado/a	97	3.8	55	4.4	42	3.2
	N.C.	8	0.3	4	0.3	4	0.3
	#Total cases	2557	2557	1256	1256	1301	1301

Figure 4.5: Combinación de casos y porcentajes dentro de las columnas

Y la misma tabla pero con los estadísticos en las filas combinando frecuencia y porcentaje...

```
as.datatable_widget(data %>%
  tab_cells(P33) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_pivot(stat_position = "outside_columns"))
```

		#Total	Sexo de la persona entrevistada		#Total	Sexo de la persona entrevistada	
		Casos	Hombre	Mujer	% casos	Hombre	Mujer
		Casos	Casos	Casos	% casos	% casos	% casos
Estado civil de la persona entrevistada	Casado/a	1388	677	711	54.3	53.9	54.7
	Soltero/a	817	455	362	32	36.2	27.8
	Viado/a	190	41	149	7.4	3.3	11.5
	Separado/a	57	24	33	2.2	1.9	2.5
	Divorciado/a	97	55	42	3.8	4.4	3.2
	N.C.	8	4	4	0.3	0.3	0.3
	#Total cases	2557	1256	1301	2557	1256	1301

Figure 4.6: Combinación de casos y porcentajes por tipo de estadístico en columnas

Y la misma tabla pero con los estadísticos en las filas combinando frecuencia y porcentaje...

```
as.datatable_widget(data %>%
  tab_cells(P33) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_pivot(stat_position = "inside_rows"))
```

			#Total	Sexo de la persona entrevistada	
				Hombre	Mujer
Estado civil de la persona entrevistada	Casado/a	Casos	1388	677	711
		% casos	54.3	53.9	54.7
	Soltero/a	Casos	817	455	362
		% casos	32	36.2	27.8
	Viado/a	Casos	190	41	149
		% casos	7.4	3.3	11.5
	Separado/a	Casos	57	24	33
		% casos	2.2	1.9	2.5
	Divorciado/a	Casos	97	55	42
		% casos	3.8	4.4	3.2
	N.C.	Casos	8	4	4
		% casos	0.3	0.3	0.3
	#Total cases	Casos	2557	1256	1301
		% casos	2557	1256	1301

Figure 4.7: Combinación de casos y porcentajes dentro de las filas

Y la misma tabla pero con los estadísticos en las filas por bloque de tipo de estadístico...

```
as.datatable_widget(data %>%
  tab_cells(P33) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_pivot(stat_position = "outside_rows"))
```

			#Total		Sexo de la persona entrevistada	
					Hombre	Mujer
Estado civil de la persona entrevistada	Casado/a	Casos	1388	677	711	
	Soltero/a	Casos	817	455	362	
	Viado/a	Casos	190	41	149	
	Separado/a	Casos	57	24	33	
	Divorciado/a	Casos	97	55	42	
	N.C.	Casos	8	4	4	
	#Total casos	Casos	2557	1256	1301	
	Casado/a	% casos	54.3	53.9	54.7	
	Soltero/a	% casos	32	36.2	27.8	
	Viado/a	% casos	7.4	3.3	11.5	
	Separado/a	% casos	2.2	1.9	2.5	
	Divorciado/a	% casos	3.8	4.4	3.2	
	N.C.	% casos	0.3	0.3	0.3	
	#Total casos	% casos	2557	1256	1301	

Figure 4.8: Combinación de casos y porcentajes, por tipo de estadístico en filas

4.2 Básica con múltiples

Vamos a realizar ahora el mismo conjunto de tablas, pero en las filas, en lugar de una variable de tipo simple, vamos a utilizar una variable de tipo múltiple. Repetimos los cruces pero cambiamos las celdas donde ahora usaremos la variable P21A con la instrucción `tab_cells(mdset(P21A01 %to% P21A03))`.

4.2.1 De frecuencias, variable múltiple y sólo absolutos

```
as.datatable_widget(data %>%
  tab_cells(mdset(P21A01 %to% P21A03)) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cases(total_row_position = "above", total_label = "Total") %>%
  tab_pivot())
```

	#Total	Sexo de la persona entrevistada	
		Hombre	Mujer
#Total	415	206	209
Medicamentos que recetan por adelantado (para que no fallen)	225	108	117
Envases que han quedado sin usar porque cambiaron el tratamiento	136	70	66
Medicamentos que decidió no tomar	82	42	40

Figure 4.9: Cruce de motivos guardar envases enteros por sexo

4.2.2 De frecuencias, variable múltiple, con porcentajes de columna

Al igual que hicimos con las tablas marginales vamos a repetir esta tabla, pero en porcentaje de columna (vertical).

```
as.datatable_widget(data %>%
  tab_cells(mdset(P21A01 %to% P21A03)) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cpct(total_row_position = "above", total_label = "Total") %>% # aquí seña
  tab_pivot())
```

	#Total	Sexo de la persona entrevistada	
		Hombre	Mujer
#Total	415	206	209
Medicamentos que recetan por adelantado (para que no fallen)	54.2	52.4	56
Envases que han quedado sin usar porque cambiaron el tratamiento	32.8	34	31.6
Medicamentos que decidió no tomar	19.8	20.4	19.1

Figure 4.10: Cruce de motivos guardar envases enteros por sexo, porcentaje de columna

4.2.3 De frecuencias, variable múltiple, con porcentajes de fila

Al igual que hicimos con las tablas marginales vamos a repetir esta tabla, pero en porcentaje de fila (horizontal).

```
as.datatable_widget(data %>%
  tab_cells(mdset(P21A01 %to% P21A03)) %>%
  tab_cols(total(), P31) %>%
  tab_stat_rpct(total_row_position = "above", total_label = "Total") %>% # aquí señalo los porcentajes
  tab_pivot())
```

	#Total	Sexo de la persona entrevistada	
		Hombre	Mujer
#Total	415	206	209
Medicamentos que recetan por adelantado (para que no fallen)	100	48	52
Envases que han quedado sin usar porque cambiaron el tratamiento	100	51.5	48.5
Medicamentos que decidió no tomar	100	51.2	48.8

Figure 4.11: Cruce de motivos guardar envases enteros por sexo, porcentaje de fila

4.2.4 De frecuencias, variable múltiple, con porcentajes total muestra

Al igual que hicimos con las tablas marginales vamos a repetir esta tabla, pero en porcentaje sobre el total de la muestra.

```
as.datatable_widget(data %>%
  tab_cells(mdset(P21A01 %to% P21A03)) %>%
  tab_cols(total(), P31) %>%
  tab_stat_tpct(total_row_position = "above", total_label = "Total") %>% # aquí seña
  tab_pivot())
```


	#Total	Sexo de la persona entrevistada	
		Hombre	Mujer
#Total	415	206	209
Medicamentos que recetan por adelantado (para que no fallen)	54.2	26	28.2
Envases que han quedado sin usar porque cambiaron el tratamiento	32.8	16.9	15.9
Medicamentos que decidió no tomar	19.8	10.1	9.6

Figure 4.12: Cruce de motivos guardar envases enteros por sexo, porcentaje sobre el total de la muestra

4.2.5 Combinaciones de los anteriores

Vamos ahora a hacer combinaciones entre ellos. Advierto que cada vez se dificulta más la tabla en su lectura. Como dije inicialmente, me decanto más por tablas sencillas y con un sólo dato.

```
as.datatable_widget(data %>%
  tab_cells(mdset(P21A01 %to% P21A03)) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_pivot(stat_position = "inside_columns"))
```

	#Total		Sexo de la persona entrevistada			
	Casos	% casos	Hombre		Mujer	
			Casos	% casos	Casos	% casos
Medicamentos que recetan por adelantado (para que no fallen)	225	54.2	108	52.4	117	56
Envases que han quedado sin usar porque cambiaron el tratamiento	136	32.8	70	34	66	31.6
Medicamentos que decidió no tomar	82	19.8	42	20.4	40	19.1
#Total cases	415	415	206	206	209	209

Figure 4.13: Combinación de casos y porcentajes dentro de las columnas

Y la misma tabla pero con los estadísticos en las filas combinando frecuencia y porcentaje...

```
as.datatable_widget(data %>%
  tab_cells(mdset(P21A01 %to% P21A03)) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_pivot(stat_position = "outside_columns"))
```

	#Total	Sexo de la persona entrevistada		#Total	Sexo de la persona entrevistada	
		Hombre	Mujer		Hombre	Mujer
	Casos	Casos	Casos	% casos	% casos	% casos
Medicamentos que recetan por adelantado (para que no fallen)	225	108	117	54.2	52.4	56
Envases que han quedado sin usar porque cambiaron el tratamiento	136	70	66	32.8	34	31.6
Medicamentos que decidió no tomar	82	42	40	19.8	20.4	19.1
#Total cases	415	206	209	415	206	209

Figure 4.14: Combinación de casos y porcentajes por tipo de estadístico en columnas

Y la misma tabla pero con los estadísticos en las filas combinando frecuencia y porcentaje...

```
as.datatable_widget(data %>%
  tab_cells(mdset(P21A01 %to% P21A03)) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_pivot(stat_position = "inside_rows"))
```

		#Total	Sexo de la persona entrevistada	
			Hombre	Mujer
Medicamentos que recetan por adelantado (para que no fallen)	Casos	225	108	117
	% casos	54.2	52.4	56
Envases que han quedado sin usar porque cambiaron el tratamiento	Casos	136	70	66
	% casos	32.8	34	31.6
Medicamentos que decidió no tomar	Casos	82	42	40
	% casos	19.8	20.4	19.1
#Total casos	Casos	415	206	209
	% casos	415	206	209

Figure 4.15: Combinación de casos y porcentajes dentro de las filas

Y la misma tabla pero con los estadísticos en las filas por bloque de tipo de estadístico...

```
as.datatable_widget(data %>%
  tab_cells(mdset(P21A01 %to% P21A03)) %>%
  tab_cols(total(), P31) %>%
  tab_stat_cases(label = "Casos") %>%
  tab_stat_cpct(label = "% casos") %>%
  tab_pivot(stat_position = "outside_rows"))
```

		Sexo de la persona entrevistada		
		#Total	Hombre	Mujer
Medicamentos que recetan por adelantado (para que no fallen)	Casos	225	108	117
Envases que han quedado sin usar porque cambiaron el tratamiento	Casos	136	70	66
Medicamentos que decidió no tomar	Casos	82	42	40
#Total cases	Casos	415	206	209
Medicamentos que recetan por adelantado (para que no fallen)	% casos	54.2	52.4	56
Envases que han quedado sin usar porque cambiaron el tratamiento	% casos	32.8	34	31.6
Medicamentos que decidió no tomar	% casos	19.8	20.4	19.1
#Total cases	% casos	415	206	209

Figure 4.16: Combinación de casos y porcentajes, por tipo de estadístico en filas

Recordamos que siempre con las múltiples existe la posibilidad de calcular los porcentajes con base respuesta en lugar de con base cuestionario (individuos). Para ello debes utilizar `tab_stat_cpct_responses()`.

4.3 Básica con estadísticos

Del mismo modo que antes utilizábamos la tabla cruzada para obtener los casos de intersección entre las categorías de columna y las categorías de fila, ahora procederemos a hacer lo mismo pero con categorías en columnas y cálculo de estadísticos básicos en otro. En definitiva, calcular las medidas estadísticas para cada grupo creado por la variable que general las categorías.

4.3.1 Cruce entre variable simple y dos estadísticos

Vamos a comenzar con las más simples, dos estadísticos (media y desviación) de una variable métrica (P3) calculados para una variable (P31) que genera dos categorías (hombre y mujer). Nótese el juego a realizar con más de 2 estadísticos con la ubicación de los mismos.

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), P31) %>%
  tab_stat_mean(label = "Media") %>%
```

```
tab_stat_sd(label = "Desviación") %>%
tab_pivot()
```

	#Total	Sexo de la persona entrevistada	
		Hombre	Mujer
Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español	Media	7.3	7.4
	Desviación	6.6	7.8

Figure 4.17: Cruce entre variable y estadísticos

Hagamos ahora su traspuesta, es decir ubiquemos en filas P31 y en columnas

```
var_lab(data$P3) = "Satisfacción"
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_rows(total(), P31) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot())
```

#Total			
#Total	Satisfacción	Medin	7.3
Sexo de la persona entrevistada	Hombre	Satisfacción	Media 7.3
	Mujer	Satisfacción	Media 7.4
#Total	Satisfacción	Desviación	7.2
Sexo de la persona entrevistada	Hombre	Satisfacción	Desviación 6.6
	Mujer	Satisfacción	Desviación 7.8

Figure 4.18: Transposición de tabla

Recordemos que los estadísticos los podemos ir moviendo a nuestra necesidad para que se organicen de una forma u otra...

Dentro de las columnas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), P31) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "inside_columns"))
```

	#Total		Sexo de la persona entrevistada			
	Media	Desviación	Hombre		Mujer	
			Media	Desviación	Media	Desviación
Satisfacción	7.3	7.2	7.3	6.6	7.4	7.8

Figure 4.19: Estadísticos dentro de columnas

Dentro de las filas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), P31) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "inside_rows"))
```


		#Total	Sexo de la persona entrevistada		
			Hombre	Mujer	
Satisfacción	Media	7.3	7.3	7.4	
	Desviación	7.2	6.6	7.8	

Figure 4.20: Estadísticos dentro de las filas

Como columnas separadas o fuera de columnas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), P31) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "outside_columns"))
```

	#Total	Sexo de la persona entrevistada		#Total	Sexo de la persona entrevistada	
	Media	Hombre	Mujer	Desviación	Hombre	Mujer
		Media	Media		Desviación	Desviación
Satisfacción	7.3	7.3	7.4	7.2	6.6	7.8

Figure 4.21: Estadísticos separados en columnas

Como filas separadas o fuera de filas...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), P31) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "outside_rows"))
```

		#Total	Sexo de la persona entrevistada	
			Hombre	Mujer
Satisfacción	Media	7.3	7.3	7.4
	Desviación	7.2	6.6	7.3

Figure 4.22: Estadísticos separados en filas

Repitamos ahora estas cuatro últimas tablas, pero en lugar de con una variable que genera grupos y de ellos se calcula la medida estadística, vamos a hacerlo con un cruce de categorías (un campo en columnas y otro en filas) y que en esos cruces, se calcule la medida estadística. Por ejemplo esta tabla me permitiría saber la media de P3 en los hombres de 18 a 25 años.

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), P31) %>%
  tab_rows(P33) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot())
```

				#Total	Sexo de la persona entrevistada	
					Hombre	Mujer
Estado civil de la persona entrevistada	Casado/a	Satisfacción	Media	7.3	7.5	7.2
	Soltero/a	Satisfacción	Media	7.2	6.9	7.4
	Viudo/a	Satisfacción	Media	7.9	7.2	8.1
	Separado/a	Satisfacción	Media	6.7	6.5	6.9
	Divorciado/a	Satisfacción	Media	7.5	8.3	6.3
	N.C.	Satisfacción	Media	18.2	7.5	29
	Casado/a	Satisfacción	Desviación	6.7	7.2	6.2
	Soltero/a	Satisfacción	Desviación	7.4	4.6	9.8
	Viudo/a	Satisfacción	Desviación	6.9	2.3	7.7
	Separado/a	Satisfacción	Desviación	2.1	2.5	1.8
Divorciado/a	Satisfacción	Desviación	9.5	12.5	2.3	
N.C.	Satisfacción	Desviación	32.3	2.4	46	

Figure 4.23: Uso de variable base (anidación)

y juguemos con la posición del cálculo estadística que ahora sí arrojará cuatro configuraciones diferentes.

La primera con los estadísticos fuera de las filas...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), P31) %>%
  tab_rows(P33) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "outside_rows")) # por defecto, sin lo ponemos muestra es
```

				#Total	Sexo de la persona entrevistada	
					Hombre	Mujer
Estado civil de la persona entrevistada	Casado/a	Satisfacción	Media	7.3	7.5	7.2
	Soltero/a	Satisfacción	Media	7.2	6.9	7.4
	Viado/a	Satisfacción	Media	7.9	7.2	8.1
	Separado/a	Satisfacción	Media	6.7	6.5	6.9
	Divorciado/a	Satisfacción	Media	7.5	8.3	6.3
	N.C.	Satisfacción	Media	18.2	7.5	29
	Casado/a	Satisfacción	Desviación	6.7	7.2	6.2
	Soltero/a	Satisfacción	Desviación	7.4	4.6	9.8
	Viado/a	Satisfacción	Desviación	6.9	2.3	7.7
	Separado/a	Satisfacción	Desviación	2.1	2.5	1.8
	Divorciado/a	Satisfacción	Desviación	9.5	12.5	2.3
	N.C.	Satisfacción	Desviación	32.3	2.4	46

Figure 4.24: Filas con estadístico fuera de filas

Los estadísticos dentro de las filas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), P31) %>%
  tab_rows(P33) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "inside_rows"))
```

				#Total	Sexo de la persona entrevistada	
					Hombre	Mujer
Estado civil de la persona entrevistada	Casado/a	Satisfacción	Media	7.3	7.5	7.2
			Desviación	6.7	7.2	6.2
	Soltero/a	Satisfacción	Media	7.2	6.9	7.4
			Desviación	7.4	4.6	9.8
	Viudo/a	Satisfacción	Media	7.9	7.2	8.1
			Desviación	6.9	2.3	7.7
	Separado/a	Satisfacción	Media	6.7	6.5	6.9
			Desviación	2.1	2.5	1.8
	Divorciado/a	Satisfacción	Media	7.5	8.3	6.3
			Desviación	9.5	12.5	2.3
	N.C.	Satisfacción	Media	18.2	7.5	29
			Desviación	32.3	2.4	46

Figure 4.25: Filas con estadístico dentro de las filas

Los estadísticos dentro de las columnas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), P31) %>%
  tab_rows(P33) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "inside_columns"))
```

			#Total		Sexo de la persona entrevistada					
					Hombre			Mujer		
			Media	Desviación	Media	Desviación	Media	Desviación	Media	Desviación
Estado civil de la persona entrevistada	Casado/a	Satisfacción	7.3	6.7	7.5	7.5	7.2	7.2	6.2	
	Soltero/a	Satisfacción	7.2	7.4	6.9		4.6	7.4	9.8	
	Viado/a	Satisfacción	7.9	6.9	7.2		2.3	8.1	7.7	
	Separado/a	Satisfacción	6.7	2.1	6.5		2.5	6.9	1.8	
	Divorciado/a	Satisfacción	7.5	9.5	8.3		12.5	6.3	2.3	
	N.C.	Satisfacción	18.2	32.3	7.5		2.4	29	46	

Figure 4.26: Filas con estadísticos dentro de las columnas

Los estadísticos fuera de las columnas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), P31) %>%
  tab_rows(P33) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "outside_columns"))
```

			#Total	Sexo de la persona entrevistada		#Total	Sexo de la persona entrevistada	
			Media	Hombre	Mujer	Desviación	Hombre	Mujer
				Media	Media		Desviación	Desviación
Estado civil de la persona entrevistada	Casado/a	Satisfacción	7.3	7.5	7.2	6.7	7.2	6.2
	Soltero/a	Satisfacción	7.2	6.9	7.4	7.4	4.6	9.8
	Viudo/a	Satisfacción	7.9	7.2	8.1	6.9	2.3	7.7
	Separado/a	Satisfacción	6.7	6.5	6.9	2.1	2.5	1.8
	Divorciado/a	Satisfacción	7.5	8.3	6.3	9.5	12.5	2.3
	N.C.	Satisfacción	18.2	7.5	29	32.3	2.4	46

Figure 4.27: Filas con estadísticos fuera de las columnas

¿Hacemos lo mismo para una variable múltiple?

4.3.1.1 Tabulación cruzada (con cálculo estadístico) y múltiples

Vamos a comenzar con las más simples, dos estadísticos (media y desviación) de una variable métrica (P3) calculados para una variable múltiple (P4_1 a P4_3) que genera categorías. Nótese el juego a realizar con más de 2 estadísticos con la ubicación de los mismos.

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), mdset(P21A01 %to% P21A03)) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot())
```


	#Total	Medicamentos que recetan por adelantado (para que no falten)	Envases que han quedado sin usar porque cambiaron el tratamiento	Medicamentos que decidió no tomar
Satisfacción	Media	7.3	7.2	6.9
	Desviación	7.2	6.4	1.5

Figure 4.28: Cruce con múltiple y estadístico (1)

Hagamos ahora su traspuesta, es decir ubiquemos en filas P21A y en columnas P3.

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_rows(total(), mdset(P21A01 %to% P21A03)) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot())
```

			#Total
#Total	Satisfacción	Media	7.3
Medicamentos que recetan por adelantado (para que no fallen)	Satisfacción	Media	7.2
Envases que han quedado sin usar porque cambiaron el tratamiento	Satisfacción	Media	6.9
Medicamentos que decidió no tomar	Satisfacción	Media	6.7
#Total	Satisfacción	Desviación	7.2
Medicamentos que recetan por adelantado (para que no fallen)	Satisfacción	Desviación	6.4
Envases que han quedado sin usar porque cambiaron el tratamiento	Satisfacción	Desviación	1.5
Medicamentos que decidió no tomar	Satisfacción	Desviación	1.7

Figure 4.29: Cruce con múltiple y estadístico (2)

Recordemos que los estadísticos los podemos ir moviendo a nuestra necesidad para que se organicen de una forma u otra...

Dentro de columnas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), mdset(P21A01 %to% P21A03)) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "inside_columns"))
```

	#Total		Medicamentos que recetan por adelantado (para que no falten)		Envases que han quedado sin usar porque cambiaron el tratamiento		Medicamentos que decidió no tomar	
	Media	Desviación	Media	Desviación	Media	Desviación	Media	Desviación
Satisfacción	7.3	7.2	7.2	6.4	6.9	1.5	6.7	1.7

Figure 4.30: Cruce con múltiple y estadístico (3)

Dentro de filas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), mdset(P21A01 %to% P21A03)) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "inside_rows"))
```

	#Total	Medicamentos que recetan por adelantado (para que no falten)	Envases que han quedado sin usar porque cambiaron el tratamiento	Medicamentos que decidió no tomar
Satisfacción Media	7.3	7.2	6.9	6.7
Desviación	7.2	6.4	1.5	1.7

Figure 4.31: Cruce con múltiple y estadístico (4)

Fuera de columnas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), mdset(P21A01 %to% P21A03)) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "outside_columns"))
```

	#Total	Medicamentos que recetan por adelantado (para que no falten)	Envases que han quedado sin usar porque cambiaron el tratamiento	Medicamentos que decidió no tomar	#Total	Medicamentos que recetan por adelantado (para que no falten)	Envases que han quedado sin usar porque cambiaron el tratamiento	Medicamentos que decidió no tomar
	Media	Media	Media	Media	Desviación	Desviación	Desviación	Desviación
Satisfacción	7.3	7.2	6.9	6.7	7.2	6.4	1.5	1.7

Figure 4.32: Cruce con múltiple y estadístico (5)

Fuera de filas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), mdset(P21A01 %to% P21A03)) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "outside_rows"))
```

	#Total	Medicamentos que recetan por adelantado (para que no fallen)	Envases que han quedado sin usar porque cambiaron el tratamiento	Medicamentos que decidió no tomar
Satisfacción Media	7.3	7.2	6.9	6.7
Desviación	7.2	6.4	1.5	1.7

Figure 4.33: Cruce con múltiple y estadístico (6)

Repitamos ahora estas cuatro últimas tablas, pero en lugar de con una variable que genera grupos y de ellos se calcula la medida estadística, vamos a hacerlo con un cruce de categorías (un campo en columnas y otro en filas) y que en esos cruces, se calcule la medida estadística. Por ejemplo esta tabla me permitiría saber la media de P3 en los hombres de 18 a 25 años.

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), mdset(P21A01 %to% P21A03)) %>%
  tab_rows(P33) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot())
```

				#Total	Medicamentos que recetan por adelantado (para que no falten)	Envases que han quedado sin usar porque cambiaron el tratamiento	Medicamentos que decidió no tomar
Estado civil de la persona entrevistada	Casado/a	Satisfacción	Media	7.3	6.7	6.7	7
	Soltero/a	Satisfacción	Media	7.2	8.2	6.8	6.4
	Viado/a	Satisfacción	Media	7.9	7.3	7.8	5
	Separado/a	Satisfacción	Media	6.7	7.8	5.3	5
	Divorciado/a	Satisfacción	Media	7.5	6.4	7.1	8.7
	N.C.	Satisfacción	Media	18.2			
	Casado/a	Satisfacción	Desviación	6.7	2	1.5	1.3
	Soltero/a	Satisfacción	Desviación	7.4	11.4	1.4	2
	Viado/a	Satisfacción	Desviación	6.9	1.7	2	
	Separado/a	Satisfacción	Desviación	2.1	0.8	0.6	
	Divorciado/a	Satisfacción	Desviación	9.5	2.1	1.2	2.3
	N.C.	Satisfacción	Desviación	32.3			

Figure 4.34: Cruce con múltiple y estadístico (7)

y juguemos con la posición del cálculo estadística que ahora sí arrojará cuatro configuraciones diferentes. La primera con los estadísticos dentro de las filas es idéntica a la anterior (por defecto).

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), mdset(P21A01 %to% P21A03)) %>%
  tab_rows(P33) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "outside_rows")) # por defecto, sin lo ponemos muestra esta opción
```

				#Total	Medicamentos que recetan por adelantado (para que no falten)	Envases que han quedado sin usar porque cambiaron el tratamiento	Medicamentos que decidió no tomar
Estado civil de la persona entrevistada	Casado/a	Satisfacción	Media	7.3	6.7	6.7	7
	Soltero/a	Satisfacción	Media	7.2	8.2	6.8	6.4
	Viado/a	Satisfacción	Media	7.9	7.3	7.8	5
	Separado/a	Satisfacción	Media	6.7	7.8	5.3	5
	Divorciado/a	Satisfacción	Media	7.5	6.4	7.1	8.7
	N.C.	Satisfacción	Media	18.2			
	Casado/a	Satisfacción	Desviación	6.7	2	1.5	1.3
	Soltero/a	Satisfacción	Desviación	7.4	11.4	1.4	2
	Viado/a	Satisfacción	Desviación	6.9	1.7	2	
	Separado/a	Satisfacción	Desviación	2.1	0.8	0.6	
	Divorciado/a	Satisfacción	Desviación	9.5	2.1	1.2	2.3
	N.C.	Satisfacción	Desviación	32.3			

Figure 4.35: Cruce con múltiple y estadístico (8)

Los estadísticos fuera de las filas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), mdset(P21A01 %to% P21A03)) %>%
  tab_rows(P33) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "inside_rows"))
```


				#Total	Medicamentos que recetan por adelantado (para que no falten)	Envases que han quedado sin usar porque cambiaron el tratamiento	Medicamentos que decidió no tomar
Estado civil de la persona entrevistada	Casado/a	Satisfacción	Media	7.3	6.7	6.7	7
			Desviación	6.7	2	1.5	1.3
	Soltero/a	Satisfacción	Media	7.2	8.2	6.8	6.4
			Desviación	7.4	11.4	1.4	2
	Viado/a	Satisfacción	Media	7.9	7.3	7.8	5
			Desviación	6.9	1.7	2	5
	Separado/a	Satisfacción	Media	6.7	7.8	5.3	5
			Desviación	2.1	0.8	0.6	
	Divorciado/a	Satisfacción	Media	7.5	6.4	7.1	8.7
			Desviación	9.5	2.1	1.2	2.3
	N.C.	Satisfacción	Media	18.2			
			Desviación	32.3			

Figure 4.36: Cruce con múltiple y estadístico (9)

Los estadísticos dentro de las columnas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), mdset(P21A01 %to% P21A03)) %>%
  tab_rows(P33) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "inside_columns"))
```

			#Total		Medicamentos que recetan por adelantado (para que no falten)		Envases que han quedado sin usar porque cambiaron el tratamiento		Medicamentos que decidió no tomar	
			Media	Desviación	Media	Desviación	Media	Desviación	Media	Desviación
Estado civil de la persona entrevistada	Casado/a	Satisfacción	7.3	6.7	6.7	2	6.7	1.5	7	1.3
	Soltero/a	Satisfacción	7.2	7.4	8.2	11.4	6.8	1.4	6.4	2
	Viado/a	Satisfacción	7.9	6.9	7.3	1.7	7.8	2	5	
	Separado/a	Satisfacción	6.7	2.1	7.8	0.8	5.3	0.6	5	
	Divorciado/a	Satisfacción	7.5	9.5	6.4	2.1	7.1	1.2	8.7	2.3
N.C.			18.2	32.3						

Figure 4.37: Cruce con múltiple y estadístico (10)

Los estadísticos fuera de las columnas ...

```
as.datatable_widget(data %>%
  tab_cells(P3) %>%
  tab_cols(total(), mdset(P21A01 %to% P21A03)) %>%
  tab_rows(P33) %>%
  tab_stat_mean(label = "Media") %>%
  tab_stat_sd(label = "Desviación") %>%
  tab_pivot(stat_position = "outside_columns"))
```

			#Total		Medicamentos que recetan por adelantado (para que no falten)		Envases que han quedado sin usar porque cambiaron el tratamiento		Medicamentos que decidió no tomar	
			Media	Media	Media	Media	Desviación	Desviación	Desviación	Desviación
Estado civil de la persona entrevistada	Casado/a	Satisfacción	7.3	6.7	6.7	7	6.7	2	1.5	1.3
	Soltero/a	Satisfacción	7.2	8.2	6.8	6.4	7.4	11.4	1.4	2
	Viado/a	Satisfacción	7.9	7.3	7.8	5	6.9	1.7	2	
	Separado/a	Satisfacción	6.7	7.8	5.3	5	2.1	0.8	0.6	
	Divorciado/a	Satisfacción	7.5	6.4	7.1	8.7	9.5	2.1	1.2	2.3
N.C.			18.2				32.3			

Figure 4.38: Cruce con múltiple y estadístico (11)

Bien, como has podido observar, el resultado no difiere cuando es múltiple a cuando es simple. Igual que hemos calculado la media y la desviación lo podemos hacer con otros estadísticos:

- media;
- desviación;
- máximo;
- mínimo;
- mediana;
- suma;
- error estándar;
- un caso especial que calcula media, desviación y el número de casos;
- y algunos otros que iremos mostrando para temas muy específicos.

Chapter 5

Tablas especiales

En esta sección nos introducimos en algunas de las características que hacen muy especial a este paquete **expss**. Iremos introduciendo cada uno de los conceptos en el punto donde corresponda, dando la explicación de caso de uso, más que su explicación técnica. El usuario que quiera conocer más sobre ello, puede acudir a la guía de referencia de todas las opciones de este paquete.

Aprenderemos a usar:

- `tab_subgroup()`, que permite hacer la tabla con una parte de los datos;
- `%nest%`, que permite anidar variables;
- `total_row_label()`, escribir textos libres en filas en la secuencia que se indique;
- `recode()`, recodificar en línea una variable (ver 6.2);
- `var_lab()`, modificar o asignar el texto de la variable;
- `val_lab()`, modificar o asignar los textos de los códigos;
- `drop_rc()`, borrar columnas o filas [`drop_empty_rows()` y `drop_empty_cols()`];
- `tab_sort_asc()`, para ordenar ascendente o descendente `tab_sort_desc()`;
- uso de `na_if`, para eliminar valores de niveles que no se desean computar en el cálculo de estadísticos;
- uso de criterios, para establecer sencillos mecanismos de filtro como complemento a otras funciones.

5.1 Subgrupos (filtros en la tabla)

Una de las primeras cuestiones que nos surge en muchas ocasiones es trabajar con subgrupos de la información original, bien porque deseamos una tabla de sólo un parte de la muestra o bien porque buscamos comparar determinados elementos en una misma tabla. Veamos las posibilidades y las situaciones de uso.

En la siguiente tabla, filtramos el banco de datos para aquellos entrevistados que han declarado tener hijos ($P34 == 1$) por un lado y los que no por otro ($P34 == 2$). Para ello usamos la instrucción `tab_subgroup()` e introducimos también la instrucción `tab_row_label()` donde escribimos una fila con texto libre. Puedes consultar en la sección ?? como se escriben las condiciones y cuáles son los operadores que puede utilizar `expss`, que su vez, tiene algunos operadores propios que en esta misma sección incluiremos.

```
as.datatable_widget(data %>%
  tab_subgroup(P34 == 1) %>%
  tab_row_label("Entrevistado con hijos") %>%
  tab_cols(total(), P31) %>%
  tab_cells(P36) %>%
  tab_stat_cases() %>%
  tab_subgroup(P34 == 2) %>%
  tab_row_label("Entrevistado sin hijos") %>%
  tab_cols(total(), P31) %>%
  tab_cells(P36) %>%
  tab_stat_cases() %>%
  tab_pivot())
```

		#Total	Sexo de la persona entrevistada	
			Hombre	Mujer
Entrevistado con hijos				
Estado de salud en general	Muy bueno	175	85	90
	Bueno	484	230	254
	Regular	125	53	72
	Malo	14	8	6
	Muy malo	3	2	1
	N.S.			
	N.C.			
	#Total cases	801	378	423
Entrevistado sin hijos				
Estado de salud en general	Muy bueno	273	151	122
	Bueno	930	490	440
	Regular	442	191	251
	Malo	72	32	40
	Muy malo	19	6	13
	N.S.	2		2
	N.C.	2	1	1
	#Total cases	1740	871	869

Figure 5.1: Tabla con subgrupos

5.2 Bases y/o con anidación

El concepto de base, no está contemplado como tal en `expss`, pero podemos utilizar las diferentes combinaciones de instrucciones para poder obtener el resultado deseado. Por ejemplo, si queremos trabajar con bases de la variable P34 (entrevistado tiene hijos o no tiene hijos), y obtener réplica de la tabla para ambas categorías, lo podíamos hacer de la forma anterior, pero hay otra posibilidad y es utilizar la opción de añadir variables anidadas en filas o utilizar de forma simultánea las instrucciones `tab_rows()` y `tab_cells()`.

En esta tabla hay algunas cosas a reseñar.

- En primer lugar `P34 == 1 | P34 == 2` es la condición de los casos con los que deseamos trabajar. No queremos contemplar los No Contesta. Se podría hacer también con el `na_if()` que veremos más adelante, pero creemos más clara esta opción.
- Por otro lado, usamos `tab_cols()`, `tab_rows()` y `tab_cells()` lo que nos va a dar una publicación curiosa que responde a nuestra necesidad.
- Por último, los códigos de P34, redefinidos al principio del script, se muestran como en una primera columna, diferenciados claramente del resto. La variable en `tab_cells()` de la que se calcula el estadístico (número de casos) se muestra como en una segunda columna, aunque realmente en el *dataframe* es sólo una. Hemos dejado fuera la opción de No Contesta aunque ya anticipamos que no tenía ningún caso.

La función `var_lab()` permite asignar a una variable (`data$P34`) una lista (expresa con la función base de R denominada vector `c()` vector) de etiquetas donde el texto de la etiqueta va entrecomillado y el valor va igualado a la etiqueta. Es una forma muy simple, a la par que práctica, de etiquetar los valores de una variable.

```
var_lab(data$P34) <- c(Sí = 1, No = 2) # los valores de P34 son etiquetados como ...
as.datatable_widget(data %>%
  tab_subgroup(P34 == 1 | P34 == 2) %>%
  tab_cols(total(), P31) %>%
  tab_rows(P34) %>%
  tab_cells(P36) %>%
  tab_stat_cases() %>%
  tab_pivot())
```

				#Total	Sexo de la persona entrevistada	
					Hombre	Mujer
Tenencia de hijos menores de 18 años	No	Estado de salud en general	Muy bueno	273	151	122
			Bueno	930	490	440
			Regular	442	191	251
			Malo	72	32	40
			Muy malo	19	6	13
			N.S.	2		2
			N.C.	2	1	1
			#Total cases	1740	871	869
	Sí	Estado de salud en general	Muy bueno	175	85	90
			Bueno	484	230	254
			Regular	125	53	72
			Malo	14	8	6
			Muy malo	3	2	1
			N.S.			
			N.C.			
			#Total cases	801	378	423

Figure 5.2: Tabla con bases 1

Si nos gusta esta opción, se le puede añadir un modificador llamado `%nest%` que nos va a permitir anidar esas bases de cálculo. El modificador `%nest%` nos permite anidar los niveles o categorías de las variables. Cualquiera de las tablas anteriores puede ser anidada en tantos niveles como se desee con la opción `%nest%`. Para que la tabla no se alargue horizontalmente, reasignamos el texto extra de la variable denominada `P3bis`. Usamos también el `recode()` para dejar tan sólo tres niveles en la variable `P3`. Más adelante en la sección 6.2 una mayor profundidad con el `recode()`.

```
data$P3bis <- recode(data$P3, 0:5 ~ 1, 6:10 ~ 2)
var_lab(data$P3bis) <- "Satisfacción"
val_lab(data$P3bis) <- c(`1.Negativa` = 1, `2.Positiva` = 2)
val_lab(data$P34) <- c(`1.Sí` = 1, `2.No` = 2)
as.datatable_widget(data %>%
  tab_subgroup(P34 == 1 | P34 == 2) %>%
  tab_cols(total(), P31) %>%
  tab_rows(P3bis %nest% P34) %>%
  tab_cells(P36) %>%
  tab_stat_cases(total_row_position = "none") %>%
  tab_pivot())
```


					#Total	Sexo de la persona entrevistada		
						Hombre	Mujer	
Satisfacción	1.Negativa	Tenencia de hijos menores de 18 años	1.Sí	Estado de salud en general	Muy bueno	35	22	13
					Bueno	117	58	59
					Regular	47	16	31
					Malo	5	3	2
					Muy malo	2	2	
					N.S.			
					N.C.			
			2.No	Estado de salud en general	Muy bueno	42	23	19
					Bueno	180	85	95
					Regular	106	41	65
					Malo	16	6	10
					Muy malo	4	2	2
					N.S.			
					N.C.			
			9	Estado de salud en general	Muy bueno			
					Bueno			
					Regular			
					Malo			
					Muy malo			
					N.S.			
					N.C.			
	2.Positiva	Tenencia de hijos menores de 18 años	1.Sí	Estado de salud en general	Muy bueno	139	63	76
					Bueno	365	170	195
					Regular	78	37	41
					Malo	9	5	4
					Muy malo	1		1
					N.S.			
					N.C.			
			2.No	Estado de salud en general	Muy bueno	227	126	101
					Bueno	746	404	342
					Regular	334	150	184
					Malo	55	25	30
					Muy malo	15	4	11
					N.S.	2		2
					N.C.	2	1	1
			9	Estado de salud en general	Muy bueno			
					Bueno			
					Regular			
					Malo			
					Muy malo			
					N.S.			
					N.C.			

Figure 5.3: Tabla con bases 2

Algo más, si añadimos la instrucción `drop_rc()`, nótese la diferencia, pues desaparecen las filas y columnas sin información.

```

data$P3bis <- recode(data$P3, 0:5 ~ 1, 6:10 ~ 2)
var_lab(data$P3bis) <- "Satisfacción"
val_lab(data$P3bis) <- c(`1.Negativa` = 1, `2.Positiva` = 2)
val_lab(data$P34) <- c(`1.Si` = 1, `2.No` = 2)
as.datatable_widget(data %>%
  tab_subgroup(P34 == 1 | P34 == 2) %>%
  tab_cols(total(), P31) %>%
  tab_rows(P3bis %nest% P34) %>%
  tab_cells(P36) %>%
  tab_stat_cases(total_row_position = "none") %>%
  tab_pivot() %>%
  drop_rc())

```

					#Total	Sexo de la persona entrevistada		
						Hombre	Mujer	
Satisfacción	1.Negativa	Tenencia de hijos menores de 18 años	1.Sí	Estado de salud en general	Muy bueno	35	22	13
					Bueno	117	58	59
					Regular	47	16	31
					Malo	5	3	2
					Muy malo	2	2	
			2.No	Estado de salud en general	Muy bueno	42	23	19
					Bueno	180	85	95
					Regular	106	41	65
					Malo	16	6	10
					Muy malo	4	2	2
	2.Positiva	Tenencia de hijos menores de 18 años	1.Sí	Estado de salud en general	Muy bueno	139	63	76
					Bueno	365	170	195
					Regular	78	37	41
					Malo	9	5	4
					Muy malo	1		1
			2.No	Estado de salud en general	Muy bueno	227	126	101
					Bueno	746	404	342
					Regular	334	150	184
					Malo	55	25	30
					Muy malo	15	4	11
					N.S.	2		2
					N.C.	2	1	1

Figure 5.4: Borrando filas y columnas vacías

5.3 Características avanzadas

Hagamos ahora lo mismo, pero combinando con un cálculo estadístico, por ejemplo nuevamente la media de P6C (atención recibida en consultas). Fijémonos

en la nueva aportación realizada. En el `tab_cells()`, se le está indicando que se trabaje con la variable `P6C`, pero que considere `NA` (valores nulos) todos aquellos que sean mayores que 5. ¿Por qué? porque los valores 8 y 9, han sido asignados al no sabe y no contesta respectivamente, y no queremos que entren en la media. Esto lo hacemos usando la función `na_if()`.

Los criterios más básicos son:

- igual -> `equals`, `eq`
- no igual -> `not_equals`, `neq`, `ne`
- mayor que -> `greater`, `gt`
- mayor o igual que -> `greater_or_equal`, `gte`, `ge`
- menor que -> `less`, `lt`
- menor o igual que -> `less_or_equal`, `lte`, `le`

Puede usarse la expresión larga o también cualquiera de las cortas. Por tanto, `na_if(P6C, gt(5))` significa trabaja con la variable `P6C` considerando `NA` los valores mayores que 5.

```
var_lab(data$P6C) <- "Valoración de la atención recibida"
data$P3bis <- recode(data$P3, 0:5 ~ 1, 6:10 ~ 2)
val_lab(data$P3bis) <- c(Negativa = 1, Positiva = 2)
val_lab(data$P34) <- c(Sí = 1, No = 2)
as.datatable_widget(data %>%
  tab_subgroup(P34 == 1 | P34 == 2) %>%
  tab_cols(total(), P31) %>%
  tab_rows(P3bis %nest% P34) %>%
  tab_cells(P6C = na_if(P6C, gt(5))) %>%
  tab_stat_mean() %>%
  tab_pivot())
```

					#Total	Sexo de la persona entrevistada	
						Hombre	Mujer
Negativa	Tenencia de hijos menores de 18 años	Si	Valoración de la atención recibida	Mean	1.2	1.1	1.4
		No	Valoración de la atención recibida	Mean	1.3	1.3	1.4
		9	Valoración de la atención recibida	Mean			
Positiva	Tenencia de hijos menores de 18 años	Si	Valoración de la atención recibida	Mean	1.1	1	1.3
		No	Valoración de la atención recibida	Mean	1.2	1.2	1.2
		9	Valoración de la atención recibida	Mean			

Figure 5.5: Usando criterios

Se puede anidar también en filas y columnas. Vamos a reducir a dos el nivel de anidación para que se lea bien la tabla.

```
data$P3bis <- recode(data$P3, 0:5 ~ 1, 6:10 ~ 2)
var_lab(data$P3bis) <- "Satisfacción"
val_lab(data$P3bis) <- c(`1.Negativa` = 1, `2.Positiva` = 2)
val_lab(data$P34) <- c(`1.Sí` = 1, `2.No` = 2)
as.datatable_widget(data %>%
  tab_cols(P31 %nest% P34) %>%
  tab_rows(P3bis %nest% P33) %>%
  tab_stat_cases(total_row_position = "none") %>%
  tab_pivot())
```

				Sexo de la persona entrevistada						
				Hombre			Mujer			
				Tenencia de hijos menores de 18 años			Tenencia de hijos menores de 18 años			
				1.Sí	2.No	9	1.Sí	2.No	9	
Satisfacción	1.Negativa	Estado civil de la persona entrevistada	Casado/a	#Total	70	73	1	70	87	1
			Sojero/a	#Total	20	63		26	74	
			Viado/a	#Total	1	9		1	19	
			Separado/a	#Total	5	5		2	4	
			Divorciado/a	#Total	5	6		6	6	
			N.C.	#Total		1			1	
	2.Positiva	Estado civil de la persona entrevistada	Casado/a	#Total	204	322	3	236	311	3
			Sojero/a	#Total	45	324	2	54	203	1
			Viado/a	#Total	3	28		5	120	3
			Separado/a	#Total	4	10		10	17	
			Divorciado/a	#Total	18	25		12	18	
			N.C.	#Total	1	1	1		2	

Figure 5.6: Anidando filas y columnas

Como caso particular en muchos casos se requieren los subtotales de los niveles de anidación. Fíjate esta tabla y trata de ver las diferencias con la anterior.

```
as.datatable_widget(data %>%
  tab_cols(P31 %nest% list(total(), P34)) %>%
  tab_rows(P36 %nest% P33) %>%
  tab_stat_cases() %>%
  tab_sort_desc() %>%
  tab_pivot())
```

Estado de salud del paciente		Bueno	Estado civil de la persona casada	Series de la primera columna									
				Resumen de datos número de 10 años					Resumen de datos número de 10 años				
				Filial	Ciudad	Filial	Ciudad	Filial	Ciudad	Filial	Ciudad	Filial	Ciudad
	Bueno	Estado civil de la persona casada	Ciudad	Filial	410	171	236	3	106	105			
			Filial	causa	410	171	236	3	106	105			
			Disminución	Filial	20	10	10	10	10	10			
			causa	20	10	10	10	10	10				
			N.C.	Filial	1	1	1	1	1	1			
			causa	1	1	1	1	1	1				
			Insuficiente	Filial	10	5	7	10	5	6			
			causa	10	5	7	10	5	6				
			Insuficiente	Filial	20	10	20	1	20	40			
			causa	20	10	20	1	20	40				
		Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10			
			Filial	causa	20	10	10	10	10	10			
			Disminución	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			N.C.	Filial	1	1	1	1	1	1			
			causa	1	1	1	1	1	1				
			Insuficiente	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			Insuficiente	Filial	20	10	20	1	20	40			
			causa	20	10	20	1	20	40				
		Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10			
			Filial	causa	20	10	10	10	10	10			
			Disminución	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			N.C.	Filial	1	1	1	1	1	1			
			causa	1	1	1	1	1	1				
			Insuficiente	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			Insuficiente	Filial	20	10	20	1	20	40			
			causa	20	10	20	1	20	40				
		Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10			
			Filial	causa	20	10	10	10	10	10			
			Disminución	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			N.C.	Filial	1	1	1	1	1	1			
			causa	1	1	1	1	1	1				
			Insuficiente	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			Insuficiente	Filial	20	10	20	1	20	40			
			causa	20	10	20	1	20	40				
		Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10			
			Filial	causa	20	10	10	10	10	10			
			Disminución	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			N.C.	Filial	1	1	1	1	1	1			
			causa	1	1	1	1	1	1				
			Insuficiente	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			Insuficiente	Filial	20	10	20	1	20	40			
			causa	20	10	20	1	20	40				
		Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10			
			Filial	causa	20	10	10	10	10	10			
			Disminución	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			N.C.	Filial	1	1	1	1	1	1			
			causa	1	1	1	1	1	1				
			Insuficiente	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			Insuficiente	Filial	20	10	20	1	20	40			
			causa	20	10	20	1	20	40				
		Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10			
			Filial	causa	20	10	10	10	10	10			
			Disminución	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			N.C.	Filial	1	1	1	1	1	1			
			causa	1	1	1	1	1	1				
			Insuficiente	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			Insuficiente	Filial	20	10	20	1	20	40			
			causa	20	10	20	1	20	40				
		Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10			
			Filial	causa	20	10	10	10	10	10			
			Disminución	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			N.C.	Filial	1	1	1	1	1	1			
			causa	1	1	1	1	1	1				
			Insuficiente	Filial	10	5	10	10	10	10			
			causa	10	5	10	10	10	10				
			Insuficiente	Filial	20	10	20	1	20	40			
			causa	20	10	20	1	20	40				
	Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10				
		Filial	causa	20	10	10	10	10	10				
		Disminución	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		N.C.	Filial	1	1	1	1	1	1				
		causa	1	1	1	1	1	1					
		Insuficiente	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		Insuficiente	Filial	20	10	20	1	20	40				
		causa	20	10	20	1	20	40					
	Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10				
		Filial	causa	20	10	10	10	10	10				
		Disminución	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		N.C.	Filial	1	1	1	1	1	1				
		causa	1	1	1	1	1	1					
		Insuficiente	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		Insuficiente	Filial	20	10	20	1	20	40				
		causa	20	10	20	1	20	40					
	Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10				
		Filial	causa	20	10	10	10	10	10				
		Disminución	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		N.C.	Filial	1	1	1	1	1	1				
		causa	1	1	1	1	1	1					
		Insuficiente	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		Insuficiente	Filial	20	10	20	1	20	40				
		causa	20	10	20	1	20	40					
	Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10				
		Filial	causa	20	10	10	10	10	10				
		Disminución	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		N.C.	Filial	1	1	1	1	1	1				
		causa	1	1	1	1	1	1					
		Insuficiente	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		Insuficiente	Filial	20	10	20	1	20	40				
		causa	20	10	20	1	20	40					
	Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10				
		Filial	causa	20	10	10	10	10	10				
		Disminución	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		N.C.	Filial	1	1	1	1	1	1				
		causa	1	1	1	1	1	1					
		Insuficiente	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		Insuficiente	Filial	20	10	20	1	20	40				
		causa	20	10	20	1	20	40					
	Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10				
		Filial	causa	20	10	10	10	10	10				
		Disminución	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		N.C.	Filial	1	1	1	1	1	1				
		causa	1	1	1	1	1	1					
		Insuficiente	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		Insuficiente	Filial	20	10	20	1	20	40				
		causa	20	10	20	1	20	40					
	Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10				
		Filial	causa	20	10	10	10	10	10				
		Disminución	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		N.C.	Filial	1	1	1	1	1	1				
		causa	1	1	1	1	1	1					
		Insuficiente	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		Insuficiente	Filial	20	10	20	1	20	40				
		causa	20	10	20	1	20	40					
	Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10				
		Filial	causa	20	10	10	10	10	10				
		Disminución	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		N.C.	Filial	1	1	1	1	1	1				
		causa	1	1	1	1	1	1					
		Insuficiente	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		Insuficiente	Filial	20	10	20	1	20	40				
		causa	20	10	20	1	20	40					
	Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10				
		Filial	causa	20	10	10	10	10	10				
		Disminución	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		N.C.	Filial	1	1	1	1	1	1				
		causa	1	1	1	1	1	1					
		Insuficiente	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		Insuficiente	Filial	20	10	20	1	20	40				
		causa	20	10	20	1	20	40					
	Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10				
		Filial	causa	20	10	10	10	10	10				
		Disminución	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		N.C.	Filial	1	1	1	1	1	1				
		causa	1	1	1	1	1	1					
		Insuficiente	Filial	10	5	10	10	10	10				
		causa	10	5	10	10	10	10					
		Insuficiente	Filial	20	10	20	1	20	40				
		causa	20	10	20	1	20	40					
	Estado civil de la persona casada	Ciudad	Filial	20	10	10	10	10	10				
		Filial	causa	20	10	10	10	10	10				
		Disminución	Filial	10	5	10	10	10	10				
		causa	10	5	10	10							

Figure 5.7: Anidando con subtotales

Efectivamente, han aparecido más columnas porque hemos anidado la columna de totales a la primera variable en lugar de usar el campo, hemos usado una lista de campos y además hemos ordenado de forma descendente respecto de la columna de total.

Pues hasta aquí una nueva sección completada. Continuamos en la sección 6 incluyendo otras características de apoyo

Chapter 6

Otras utilidades y tablas especiales

Continuamos en esta sección aportando sentido y contenido al trabajo con tablas. En esta sección, introducimos la utilidad de disponer del plan de códigos y más en profundidad, trabajamos la utilidad de otros elementos básicos: cuadros, recodificación, ponderación, subtotales y NETS.

6.1 Plan de códigos

De forma muy sencilla, podemos obtener de nuestro fichero de datos lo que se denomina el *codeplan* (?) o libro de códigos. **expss** tiene una orden directa para ello. Este libro de códigos nos permitirá de forma muy sencilla conocer de primera mano, con mayor o menor precisión, de forma exploratoria el contenido de nuestro banco de datos o en definitiva la codificación que se ha utilizado.

Cuatro elementos de información son los más importantes en la creación de un plan de códigos (?):

- Conocer los textos de etiquetado de las variables en el cuestionario (ítem, escala de respuesta, instrucción del entrevistador, etc.) y conocer su caracterización en el archivo de datos a partir de su posición, longitud, nombre de campo, etc.;
- Propiedades del campo de datos numéricos (ancho, decimales, alfa (carácter), o numérico);
- Definición de los valores de escala de respuesta, filtros, rechazos, no sabe, no contesta...;
- Conocimiento de las variables de clasificación

La siguiente sintaxis nos permite alcanzar este objetivo con poco esfuerzo y como punto de partida para el siguiente punto de recodificación.

```
as.datatable_widget(info(data, frequencies = F, max_levels = 10))
```

Buscador									
Nombre									
Id	Nombre	Clase	Longitud	NA	Distancia	Label	MinDist	MaxDist	Min
1	00001	MedidaLancos	2007	2007	0	1	1000-1000	1000	1000
2	00002	MedidaLancos	2007	2007	0	2	1000-1000	1000	1000
3	00003	MedidaLancos	2007	2007	0	3	1000-1000	1000	1000
4	00004	MedidaLancos	2007	2007	0	4	1000-1000	1000	1000
5	00005	MedidaLancos	2007	2007	0	5	1000-1000	1000	1000
6	00006	MedidaLancos	2007	2007	0	6	1000-1000	1000	1000
7	00007	MedidaLancos	2007	2007	0	7	1000-1000	1000	1000
8	00008	MedidaLancos	2007	2007	0	8	1000-1000	1000	1000
9	00009	MedidaLancos	2007	2007	0	9	1000-1000	1000	1000
10	00010	MedidaLancos	2007	2007	0	10	1000-1000	1000	1000
11	00011	MedidaLancos	2007	2007	0	11	1000-1000	1000	1000
12	00012	MedidaLancos	2007	2007	0	12	1000-1000	1000	1000
13	00013	MedidaLancos	2007	2007	0	13	1000-1000	1000	1000
14	00014	MedidaLancos	2007	2007	0	14	1000-1000	1000	1000
15	00015	MedidaLancos	2007	2007	0	15	1000-1000	1000	1000
16	00016	MedidaLancos	2007	2007	0	16	1000-1000	1000	1000
17	00017	MedidaLancos	2007	2007	0	17	1000-1000	1000	1000
18	00018	MedidaLancos	2007	2007	0	18	1000-1000	1000	1000
19	00019	MedidaLancos	2007	2007	0	19	1000-1000	1000	1000
20	00020	MedidaLancos	2007	2007	0	20	1000-1000	1000	1000
21	00021	MedidaLancos	2007	2007	0	21	1000-1000	1000	1000
22	00022	MedidaLancos	2007	2007	0	22	1000-1000	1000	1000
23	00023	MedidaLancos	2007	2007	0	23	1000-1000	1000	1000
24	00024	MedidaLancos	2007	2007	0	24	1000-1000	1000	1000
25	00025	MedidaLancos	2007	2007	0	25	1000-1000	1000	1000
26	00026	MedidaLancos	2007	2007	0	26	1000-1000	1000	1000
27	00027	MedidaLancos	2007	2007	0	27	1000-1000	1000	1000
28	00028	MedidaLancos	2007	2007	0	28	1000-1000	1000	1000
29	00029	MedidaLancos	2007	2007	0	29	1000-1000	1000	1000
30	00030	MedidaLancos	2007	2007	0	30	1000-1000	1000	1000
31	00031	MedidaLancos	2007	2007	0	31	1000-1000	1000	1000
32	00032	MedidaLancos	2007	2007	0	32	1000-1000	1000	1000
33	00033	MedidaLancos	2007	2007	0	33	1000-1000	1000	1000
34	00034	MedidaLancos	2007	2007	0	34	1000-1000	1000	1000
35	00035	MedidaLancos	2007	2007	0	35	1000-1000	1000	1000
36	00036	MedidaLancos	2007	2007	0	36	1000-1000	1000	1000
37	00037	MedidaLancos	2007	2007	0	37	1000-1000	1000	1000
38	00038	MedidaLancos	2007	2007	0	38	1000-1000	1000	1000
39	00039	MedidaLancos	2007	2007	0	39	1000-1000	1000	1000
40	00040	MedidaLancos	2007	2007	0	40	1000-1000	1000	1000
41	00041	MedidaLancos	2007	2007	0	41	1000-1000	1000	1000
42	00042	MedidaLancos	2007	2007	0	42	1000-1000	1000	1000
43	00043	MedidaLancos	2007	2007	0	43	1000-1000	1000	1000
44	00044	MedidaLancos	2007	2007	0	44	1000-1000	1000	1000
45	00045	MedidaLancos	2007	2007	0	45	1000-1000	1000	1000
46	00046	MedidaLancos	2007	2007	0	46	1000-1000	1000	1000
47	00047	MedidaLancos	2007	2007	0	47	1000-1000	1000	1000
48	00048	MedidaLancos	2007	2007	0	48	1000-1000	1000	1000
49	00049	MedidaLancos	2007	2007	0	49	1000-1000	1000	1000
50	00050	MedidaLancos	2007	2007	0	50	1000-1000	1000	1000
51	00051	MedidaLancos	2007	2007	0	51	1000-1000	1000	1000
52	00052	MedidaLancos	2007	2007	0	52	1000-1000	1000	1000
53	00053	MedidaLancos	2007	2007	0	53	1000-1000	1000	1000
54	00054	MedidaLancos	2007	2007	0	54	1000-1000	1000	1000
55	00055	MedidaLancos	2007	2007	0	55	1000-1000	1000	1000
56	00056	MedidaLancos	2007	2007	0	56	1000-1000	1000	1000
57	00057	MedidaLancos	2007	2007	0	57	1000-1000	1000	1000
58	00058	MedidaLancos	2007	2007	0	58	1000-1000	1000	1000
59	00059	MedidaLancos	2007	2007	0	59	1000-1000	1000	1000
60	00060	MedidaLancos	2007	2007	0	60	1000-1000	1000	1000
61	00061	MedidaLancos	2007	2007	0	61	1000-1000	1000	1000
62	00062	MedidaLancos	2007	2007	0	62	1000-1000	1000	1000
63	00063	MedidaLancos	2007	2007	0	63	1000-1000	1000	1000
64	00064	MedidaLancos	2007	2007	0	64	1000-1000	1000	1000
65	00065	MedidaLancos	2007	2007	0	65	1000-1000	1000	1000
66	00066	MedidaLancos	2007	2007	0	66	1000-1000	1000	1000
67	00067	MedidaLancos	2007	2007	0	67	1000-1000	1000	1000
68	00068	MedidaLancos	2007	2007	0	68	1000-1000	1000	1000
69	00069	MedidaLancos	2007	2007	0	69	1000-1000	1000	1000
70	00070	MedidaLancos	2007	2007	0	70	1000-1000	1000	1000
71	00071	MedidaLancos	2007	2007	0	71	1000-1000	1000	1000
72	00072	MedidaLancos	2007	2007	0	72	1000-1000	1000	1000
73	00073	MedidaLancos	2007	2007	0	73	1000-1000	1000	1000
74	00074	MedidaLancos	2007	2007	0	74	1000-1000	1000	1000
75	00075	MedidaLancos	2007	2007	0	75	1000-1000	1000	1000
76	00076	MedidaLancos	2007	2007	0	76	1000-1000	1000	1000
77	00077	MedidaLancos	2007	2007	0	77	1000-1000	1000	1000
78	00078	MedidaLancos	2007	2007	0	78	1000-1000	1000	1000
79	00079	MedidaLancos	2007	2007	0	79	1000-1000	1000	1000
80	00080	MedidaLancos	2007	2007	0	80	1000-1000	1000	1000
81	00081	MedidaLancos	2007	2007	0	81	1000-1000	1000	1000
82	00082	MedidaLancos	2007	2007	0	82	1000-1000	1000	1000
83	00083	MedidaLancos	2007	2007	0	83	1000-1000	1000	1000
84	00084	MedidaLancos	2007	2007	0	84	1000-1000	1000	1000
85	00085	MedidaLancos	2007	2007	0	85	1000-1000	1000	1000
86	00086	MedidaLancos	2007	2007	0	86	1000-1000	1000	1000
87	00087	MedidaLancos	2007	2007	0	87	1000-1000	1000	1000
88	00088	MedidaLancos	2007	2007	0	88	1000-1000	1000	1000
89	00089	MedidaLancos	2007	2007	0	89	1000-1000	1000	1000
90	00090	MedidaLancos	2007	2007	0	90	1000-1000	1000	1000
91	00091	MedidaLancos	2007	2007	0	91	1000-1000	1000	1000
92	00092	MedidaLancos	2007	2007	0	92	1000-1000	1000	1000
93	00093	MedidaLancos	2007	2007	0	93	1000-1000	1000	1000
94	00094	MedidaLancos	2007	2007	0	94	1000-1000	1000	1000
95	00095	MedidaLancos	2007	2007	0	95	1000-1000	1000	1000
96	00096	MedidaLancos	2007	2007	0	96	1000-1000	1000	1000
97	00097	MedidaLancos	2007	2007	0	97	1000-1000	1000	1000
98	00098	MedidaLancos	2007	2007	0	98	1000-1000	1000	1000
99	00099	MedidaLancos	2007	2007	0	99	1000-1000	1000	1000
100	00100	MedidaLancos	2007	2007	0	100	1000-1000	1000	1000

Los modificadores de la función `info()`, permiten ir desde el listado más completo (el mostrado) hasta la supresión de elementos como las frecuencias o los descriptivos básicos, al tiempo que agilizan su cálculo.

```
as.datatable_widget(info(data, stats = FALSE, frequencies = FALSE))
```

6.1. PLAN DE CÓDIGOS

Show10entries

Search:

	Name	Class	Length	NotNA	NA	Distincts	Label	Value Labels
1	ESTU	labelled,numeric	2557	2557	0	1		3192-3192
2	CUES	numeric	2557	2557	0	2557		
3	CCAA	labelled,numeric	2557	2557	0	19	Comunidad autónoma	Andalucía=1, Aragón=2, Asturias (Principado de)=3, Baleares (Illes)=4, Canarias=5, Cantabria=6, Castilla-La Mancha=7, Castilla y León=8, Cataluña=9, Comunitat Valenciana=10, Extremadura=11, Galicia=12, Madrid (Comunidad de)=13, Murcia (Región de)=14, Navarra (Comunidad Foral de)=15, País Vasco=16, Rioja (La)=17, Ceuta (Ciudad Autónoma de)=18, Melilla (Ciudad Autónoma de)=19
4	PROV	labelled,numeric	2557	2557	0	51	Provincia	Araha/Alava=1, Albacete=2, Alicante/Alcant=3, Almería=4, Avila=5, Badajoz=6, Baleares (Illes)=7, Barcelona=8, Burgos=9, Cáceres=10, Cádiz=11, Castellón/Castelló=12, Ciudad Real=13, Córdoba=14, Coruña (A)=15, Cuenca=16, Girona=17, Granada=18, Guadalajara=19, Gipuzkoa=20, Huelva=21, Huesca=22, Jaén=23, León=24, Lleida=25, Rioja (La)=26, Lago=27, Madrid=28, Málaga=29, Murcia=30, Navarra=31, Ourense=32, Asturias=33, Palencia=34, Palmas (Las)=35, Pontevedra=36, Salamanca=37, Santa Cruz de Tenerife=38, Cantabria=39, Segovia=40, Sevilla=41, Soria=42, Tarragona=43, Teruel=44, Toledo=45, Valencia/Valencia=46, Valladolid=47, Bizkaia=48, Zamora=49, Zaragoza=50, Ceuta=51, Melilla=52
5	MUN	labelled,numeric	2557	2557	0	52	Municipio	Max.<=100.000 hab. no capitales de C.A o provincia=0, Ceuta / Melilla=1, Albacete=3, Algeciras=4, Alkai de Henares=5, Alcobendas=6, Alcorcón=7, Cádiz=12, Almería / Bursakallo=13, Alicante/Alcant=14, Badajoz / Badajoz=15, Cartagena / Palmas de Gran Canaria (Las)=16, Ávila / Barcelona=19, Bilbao / Jerez de la Frontera=20, Córdoba=21, La Laguna=23, Gijón=24, Toledo=26, Lago=28, Coruña (A) / Murcia=30, Ciudad Real=34, Cáceres=37, Pontevedra / Santa Cruz de Tenerife / Dos Hermanas=38, Castellón de la Plana/Castelló de la Plana / Palma de Maior=40, Huelva=41, Oviedo=44, Jaén=50, Ourense=54, Vigo=57, Fuahbreada=58, Burgos / Vitoria-Gasteiz=59, Elche/Elx / Getúfe=65, Málaga=67, Donostia/San Sebastián / Marbella=69, Leganés=74, Santander=75, Cuenca / Santiago de Compostela=78, Girona / Madrid=79, Mérida=83, Granada=87, León / Logroño=89, Sevilla=91, Móstoles=92, Hospital de Llobregat=101, Parla=106, Lleida / Palencia=120, Mataró=121, Reus=123, Huesca=125, Guadalajara=130, Tarragona / Torrejón de Ardo=148, Toledo=168, Soria=173, Valladolid=186, Sabadell=187, Segovia=194, Pamplona/Iruña=201, Teruel=216, Santa Coloma de Gramenet=245, Valencia=250, Salamanca=274, Zamora=275, Terrassa=279, Zaragoza=297
6	TAMUNI	labelled,numeric	2557	2557	0	7	Tamaño de municipio	Menos o igual a 2.000 habitantes=1, 2.001 a 10.000 habitantes=2, 10.001 a 50.000 habitantes=3, 50.001 a 100.000 habitantes=4, 100.001 a 400.000 habitantes=5, 400.001 a 1.000.000 habitantes=6, Más de 1.000.000 habitantes=7
7	CAPITAL	labelled,numeric	2557	2557	0	3	Capital	Capital de CCAA=1, Capital de provincia=2, Otros municipios=3
8	DISTR	labelled,numeric	2557	2557	0	1		Anonimizado=0
9	SECCION	labelled,numeric	2557	2557	0	1		Anonimizado=0
10	ENTREV	labelled,numeric	2557	2556	1	2		Anonimizado=0

Showing 1 to 10 of 190 entries

Previous12345...19Next

Conociendo el libro de códigos, accedamos al apartado de recodificación. Si quieres más información sobre el comando `info()` usa la función `help(topic = info)` o también `?expss:info`.

6.2 Recodificar

Aunque ya en la sección anterior hicimos una breve incursión en el uso de la recodificación, el paquete `expss` dispone de una serie de opciones muy interesantes acerca de esta funcionalidad que usamos mucho en nuestro trabajo diario. La función `recode()` cambia la codificación de una variable en el contexto que se utiliza. Puede ser usada también para reorganizar o consolidar los valores de una variable existente en función de las condiciones. El diseño de esta función está inspirada en la utilidad `RECODE` de SPSS. El usuario facilita una secuencia de recodificaciones proporcionadas en forma de fórmulas.

Por ejemplo, `1:2 ~ 1` significa que todos los valores 1 y 2 se reemplazarán con 1. Cada valor se recodificará solo una vez, es decir, sea realiza una única ‘pasada’ por el registro empezando por el 1 y acabando por el N.

Dos formas de uso:

- Si `recode()` se usa como funcionalidad diferenciada, en este proceso de asignación aquellos valores que no cumplan ninguna condición permanecen sin cambios.
- Si `recode()` se usa dentro de una tabla, los valores de recodificación (...) que no cumplen ninguna condición serán reemplazados por NA.

Se pueden usar valores o condiciones lógicas más sofisticadas y funciones como condición. Hay varias funciones especiales para su uso como criterios; para más detalles, consulte los criterios `??` en su sección `??`.

El uso común se parece a este: `recode(x, 1:2 ~ -1, 3 ~ 0, 4:5 ~ 1, 99 ~ NA)`. Se puede observar que a los valores originales 1 y 2 se les imputa un -1, al 3 un 0, y a los valores 4 y 5 se les imputa un 1, el 99 se convierte en NA (valor perdido).

Para más información, ver detalles y ejemplos a continuación. Te dejamos los ejemplos del autor que ilustran muy bien las posibilidades de esta funcionalidad. Se reproducen los ejemplos de recodificación extraídos del manual de SPSS. Se utilizan datos ficticios generados en línea.

```
# RECODE V1 TO V3 (0=1) (1=0) (2, 3=-1) (9=9)
# (ELSE=SYSMIS)
v1 = c(0, 1, 2, 3, 9, 10) # se crea la variable
v1
```

```
## [1] 0 1 2 3 9 10
```

```
recode(v1) = c(0 ~ 1, 1 ~ 0, 2:3 ~ -1, 9 ~ 9, TRUE ~ NA)
v1
```

```
## [1] 1 0 -1 -1 9 NA
```

```
# RECODE QVAR(1 THRU 5=1)(6 THRU 10=2)(11 THRU
# HI=3)(ELSE=0).
```

```
qvar = c(1:20, 97, NA, NA)
recode(qvar, 1 %thru% 5 ~ 1, 6 %thru% 10 ~ 2, 11 %thru% hi ~
  3, TRUE ~ 0)
```

```
## [1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 0 0
```

```
# the same result
```

```
recode(qvar, 1 %thru% 5 ~ 1, 6 %thru% 10 ~ 2, ge(11) ~ 3,
  TRUE ~ 0)
```

```
## [1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 0 0
```

```
# RECODE STRNGVAR ('A', 'B', 'C'='A')('D', 'E',
# 'F'='B')(ELSE=' ').
```

```
strngvar = LETTERS
recode(strngvar, c("A", "B", "C") ~ "A", c("D", "E", "F") ~
  "B", TRUE ~ " ")
```

```
## [1] "A" "A" "A" "B" "B" "B" " " " " " " " " " " " " " "
```

```
## [14] " " " " " " " " " " " " " " " " " " " " " " " "
```

```
# recode in place. Note that we recode only first six
# letters
```

```
recode(strngvar) = c(c("A", "B", "C") ~ "A", c("D", "E",
  "F") ~ "B")
strngvar
```

```
## [1] "A" "A" "A" "B" "B" "B" "G" "H" "I" "J" "K" "L" "M"
```

```
## [14] "N" "O" "P" "Q" "R" "S" "T" "U" "V" "W" "X" "Y" "Z"
```

```
# RECODE AGE (MISSING=9) (18 THRU HI=1) (0 THRU 18=0)
# INTO VOTER.
```

```
age = c(NA, 2:40, NA)
voter = recode(age, NA ~ 9, 18 %thru% hi ~ 1, 0 %thru% 18 ~
  0)
voter
```



```
## LABEL: Liking
## VALUES:
## 1, 1, 1, 4, 7, 7, 7, 99
## VALUE LABELS:
## 1 Disgusting/Very Poor/Poor
## 4 So-so
## 7 Good/Very good/Excellent
## 99 Hard to say
```

```
# 'ifs' examples
```

```
a = 1:5
```

```
b = 5:1
```

```
a
```

```
## [1] 1 2 3 4 5
```

```
b
```

```
## [1] 5 4 3 2 1
```

```
ifs(b > 3 ~ 1) # c(1, 1, NA, NA, NA)
```

```
## [1] 1 1 NA NA NA
```

```
ifs(b > 3 ~ 1, TRUE ~ 3) # c(1, 1, 3, 3, 3)
```

```
## [1] 1 1 3 3 3
```

```
ifs(b > 3 ~ 1, a > 4 ~ 7, TRUE ~ 3) # c(1, 1, 3, 3, 7)
```

```
## [1] 1 1 3 3 7
```

```
ifs(b > 3 ~ a, TRUE ~ 42) # c(1, 2, 42, 42, 42)
```

```
## [1] 1 2 42 42 42
```

El `recode()` puede ser utilizado como funcionalidad separada y con asignación o dentro de la definición de una variable. Por ejemplo:

```
v1 <- c(1, 2, 3, 2, 1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1)
```

```
v2 <- v1
```

```
v1
```

```
## [1] 1 2 3 2 1 2 3 4 5 6 5 4 3 2 1
```

```
v2
```

```
## [1] 1 2 3 2 1 2 3 4 5 6 5 4 3 2 1
```

```
v1 <- recode(v1, 1:5 ~ 1, 6:10 ~ 2)
mean(v1, na.rm = TRUE)
```

```
## [1] 1.066667
```

```
mean(recode(v2, 1:5 ~ 1, 6:10 ~ 2), na.rm = TRUE)
```

```
## [1] 1.066667
```

```
v1
```

```
## [1] 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1
```

```
v2
```

```
## [1] 1 2 3 2 1 2 3 4 5 6 5 4 3 2 1
```

Nótese que la diferencia estriba en que mientras que al finalizar el proceso `v1` tiene solo valores 1 y 2 recodificados, `v2` (copia de `v1`) mantiene los valores originales.

6.3 Cuadros

En ocasiones es interesante reproducir algún tipo de cuadro que se ha presentado al entrevistado (baterías de ítems, cuadros, rejillas o *grids*, grillas o tablas de ítems, son sinónimos). Por ejemplo la pregunta P9 del cuestionario nos presenta un cuadro en el que hay hasta 7 ítems valorados de 1 a 10, con los valores 98 y 99 como NS y NC respectivamente. Así que vamos a procesar esas tablas como cuadros.

Nuestro script es el siguiente ...

```
as.datatable_widget(data %>%
  tab_cells(P901 %to% P907) %>%
  tab_stat_cpct() %>%
  tab_pivot())
```

Los cuidados y la atención recibida del personal médico	1 Tratamiento insatisfactorio	0.6
	2	0.2
	3	0.9
	4	2.1
	5	6.8
	6	9.9
	7	18.4
	8	23.9
	9	15.1
	10 Tratamiento satisfactorio	18.5
	N.S.	1.3
	N.C.	0.2
	#Evaluaciones	2597
Los cuidados y la atención recibida del personal de enfermería	1 Tratamiento insatisfactorio	0.4
	2	0.4
	3	0.4
	4	1.4
	5	6.1
	6	10.4
	7	17.4
	8	26.1
	9	14.8
	10 Tratamiento satisfactorio	16.6
	N.S.	3.9
	N.C.	0.2
	#Evaluaciones	2597
La confianza y seguridad que transmite el personal médico	1 Tratamiento insatisfactorio	1.1
	2	0.7
	3	0.9
	4	2.1
	5	6.5
	6	9.2
	7	16.4
	8	23.5
	9	16.1
	10 Tratamiento satisfactorio	21.2
	N.S.	1.8
	N.C.	0.4
	#Evaluaciones	2597
La confianza y seguridad que transmite el personal de enfermería	1 Tratamiento insatisfactorio	0.4
	2	0.5
	3	0.7
	4	1.4
	5	6.2
	6	9.5
	7	18.7
	8	24.4
	9	14.1
	10 Tratamiento satisfactorio	17.6
	N.S.	6.1
	N.C.	0.2
	#Evaluaciones	2597
El tiempo dedicado por el médico o la médica a cada enfermo o enferma	1 Tratamiento insatisfactorio	1.4
	2	1.7
	3	2.6
	4	4.0
	5	11.5
	6	13.6
	7	17.1
	8	18.5
	9	11.5
	10 Tratamiento satisfactorio	15.1
	N.S.	2.1
	N.C.	0.2
	#Evaluaciones	2597
El conocimiento del historial y seguimiento de los problemas de salud de cada usuario o usuaria	1 Tratamiento insatisfactorio	1.6
	2	1
	3	1.5
	4	3.7
	5	7.4
	6	10.7
	7	16.5
	8	21.2
	9	15
	10 Tratamiento satisfactorio	18.3
	N.S.	4.7
	N.C.	0.4
	#Evaluaciones	2597
La información recibida sobre sus problemas de salud	1 Tratamiento insatisfactorio	1.4
	2	1.1
	3	1.7
	4	3.1
	5	7.4
	6	10.4
	7	17
	8	21.5
	9	15
	10 Tratamiento satisfactorio	18.3
	N.S.	2.7
	N.C.	0.4
	#Evaluaciones	2597

Figure 6.1: Cuadros de baterías

Esta sería la salida lógica que damos a la tabla. Sin embargo utilizando algunos pequeños trucos, podemos presentarlo así. Seguro nuestro cliente final está más contento ...

```
val_lab(data$P901) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
val_lab(data$P902) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
val_lab(data$P903) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
val_lab(data$P904) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
val_lab(data$P905) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
val_lab(data$P906) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
val_lab(data$P907) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)

as.datatable_widget(data %>%
  tab_cells(`|` = unvr(P901)) %>%
  tab_stat_cpct(label = var_lab(P901), total_row_position = "none") %>%
  tab_stat_mean_sd_n(label = "P901") %>%
  tab_cells(`|` = unvr(P902)) %>%
  tab_stat_cpct(label = var_lab(P902), total_row_position = "none") %>%
  tab_stat_mean_sd_n(label = "P901") %>%
  tab_cells(`|` = unvr(P903)) %>%
  tab_stat_cpct(label = var_lab(P903), total_row_position = "none") %>%
  tab_stat_mean_sd_n(label = "P901") %>%
  tab_cells(`|` = unvr(P904)) %>%
  tab_stat_cpct(label = var_lab(P904), total_row_position = "none") %>%
  tab_stat_mean_sd_n(label = "P901") %>%
  tab_cells(`|` = unvr(P905)) %>%
  tab_stat_cpct(label = var_lab(P905), total_row_position = "none") %>%
  tab_stat_mean_sd_n() %>%
  tab_cells(`|` = unvr(P906)) %>%
  tab_stat_cpct(label = var_lab(P906), total_row_position = "none") %>%
  tab_stat_mean_sd_n() %>%
  tab_cells(`|` = unvr(P907)) %>%
```

```

tab_stat_cpct(label = var_lab(P907), total_row_position = "none") %>%
tab_stat_mean_sd_n() %>%
tab_pivot(stat_position = "inside_columns") %>%
t()

```

	1	2	3	4	5	6	7	8	9	10	N.S.	N.C.	Mean	Std. dev.	Unw. valid N
#Total Los cuidados y la atención recibida del personal médico	0.6	0.2	0.9	2.1	6.8	9.9	18.4	25.9	15.1	18.5	1.3	0.2			
P901													9.1	11.2	2557
Los cuidados y la atención recibida del personal de enfermería	0.4	0.4	0.4	1.4	6.1	10.4	17.4	26.1	14.8	16.6	5.9	0.2			
P901.1													13.2	21.6	2557
La confianza y seguridad que transmite el personal médico	1.1	0.7	0.9	2.1	6.5	9.2	16.4	23.5	16.1	21.2	1.8	0.4			
P901.2													9.7	13.2	2557
La confianza y seguridad que transmite el personal de enfermería	0.4	0.5	0.7	1.4	6.2	9.3	18.7	24.4	14.1	17.8	6.1	0.2			
P901.3													13.5	22.1	2557
El tiempo dedicado por el médico o la médica a cada enfermo o enferma	1.4	1.7	2.6	4.8	11.5	13.6	17.1	18.3	11.5	15.1	2.1	0.2			
													9.2	13.8	2557
El conocimiento del historial y seguimiento de los problemas de salud de cada usuario o usuaria	1.6	1	1.5	3.7	7.4	10.7	16.5	21.2	13	18.3	4.7	0.4			
.1													12	19.9	2557
La información recibida sobre su problema de salud	1.4	1.1	1.7	3.1	7.4	10.4	17	21.5	15	18.3	2.7	0.4			
													10.4	16	2557

Figure 6.2: Cuadros de baterías modificado

¿Qué hemos hecho?, hemos limpiado el texto de la variable utilizando la función `"|"=unvr()` y ese mismo texto extra de la variable `var_lab()` se lo hemos asignado al estadístico con `label`. De esta forma el resultado es el que ves. Desafortunadamente, hay un pequeño bug del que está informado el autor, de no poder situar la media y/o la desviación típica en la misma fila. Para hacerlo hay que unir dos tablas. Como esperamos esté resuelto en breve, no damos la solución por no complicar más la salida.

6.4 Ponderación

Otro de los aspectos fundamentales en la investigación de mercados es la ponderación. Lo primero que debemos entender es el propio concepto de ponderación. En definitiva, es hacer que cada registro (% de casos o casos o estadísticos) en lugar de contar como un caso (frecuencia 1), cuente como n casos, siendo n el valor de otro campo (indicado en `weight()` -peso-) del marco de datos. Este peso ha sido obtenido por un procedimiento llamado equilibrio o *raking* (?).

Aquí muestro la tabla, sin ponderar ...

```
as.datatable_widget(data %>%
  tab_cols(total(), P3) %>%
  tab_cells(mdset(P21A01 %to% P21A03)) %>%
  tab_stat_cpct() %>%
  tab_pivot())
```

	#Total	Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español										N.S.	N.C.
		1 Muy insatisfecho/a	2	3	4	5	6	7	8	9	10 Muy satisfecho/a		
Medicamentos que recetan por adelantado (para que no fallen)	54.2	83.3	80	25	68.8	51.9	47.9	50	57.9	61.3	58.3	100	
Enfases que han quedado sin usar porque cambiaron el tratamiento	32.8			50	25	34.6	32.4	39.6	30.5	32.3	25		
Medicamentos que decidió no tomar	19.8	16.7	20	25	12.5	19.2	26.8	19.8	15.8	22.6	16.7		
#Total cases	415	6	5	8	16	52	71	106	95	31	24	1	

Figure 6.3: Uso de ponderación

El estudio del CIS que estamos trabajando tiene una variable denominada PESO que contiene el coeficiente de ponderación para adaptarse a la población real española. Aquí dejamos la anterior tabla, pero ponderada. Véase las diferencias entre todos los valores.

```
as.datatable_widget(data %>%
  tab_cols(total(), P3) %>%
  tab_weight(PESO) %>%
  tab_cells(mdset(P21A01 %to% P21A03)) %>%
  tab_stat_cpct() %>%
  tab_pivot())
```

	#Total	Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español										N.S.	N.C.
		1 Muy insatisfecho/a	2	3	4	5	6	7	8	9	10 Muy satisfecho/a		
Medicamentos que recetan por adelantado (para que no fallen)	54.7	80.3	88.6	19.8	63.6	55.5	44.5	52.5	57.1	64.3	66.4	100	
Envases que han quedado sin usar porque cambiaron el tratamiento	31			56.2	27.5	31.9	31.9	37.7	27.6	30	19.8		
Medicamentos que decidió no tomar	21.2	19.7	11.4	24	17.3	18.3	30.1	19.8	18.6	25.7	13.8		
#Total cases	415	6	5	8	16	52	71	106	95	31	24	1	

Figure 6.4: Uso de ponderación con multi respuesta

Aunque este manual se refiere únicamente a tablas, es bastante habitual obtener en toda ponderación la denominada eficiencia de la misma. Este análisis lo realizamos utilizando R como calculadora. Podemos hacer una análisis de la variable PESO y de su eficiencia.

```
mean.peso <- mean(data$PESO, na.rm = TRUE)
sd.peso <- sd(data$PESO, na.rm = TRUE)
ratio <- sd.peso/mean.peso
eficiencia <- (1/(1 + (ratio^2))) * 100
eficiencia <- round(eficiencia, 2)
```

Puede observarse como la eficiencia de la ponderación es del 76.79 %.

6.5 Subtotales y NETS

Otra de las funcionalidades básicas en nuestro trabajo de análisis es el uso de subtotales y/o netos. El objetivo de ambas funciones es reagrupar los códigos de una determinada variable, permitiendo observar acumulados de frecuencia. El paquete `expss` hace una diferenciación entre ambos que mostraremos seguidamente.

6.5.1 Subtotales

El uso de `tab_subtotal_rows()` o `tab_subtotal_cols()` o `tab_subtotal_cells()` añade subtotales a un conjunto de categorías de la variable sobre la que se

aplique. Si se introduce un texto se utilizará el mismo, pero si no, se añade la palabra TOTAL. Debes tener en cuenta que si las agrupaciones de categorías que realizas se solapan, también se solaparán los recuentos en el cálculo de subtotales. Estos subtotales pueden ser aplicados a las variables del banco de datos.

```
val_lab(data$P901) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
as.datatable_widget(data %>%
  tab_cols(total(), P31) %>%
  tab_cells(P901) %>%
  tab_subtotal_cells(`NO SATISFACTORIO` = 1:5, 6:8, SATISFACTORIO = 9:10) %>%
  tab_stat_cpct() %>%
  tab_pivot())
```

	#Total	Sexo de la persona entrevistada		
		Hombre	Mujer	
Los cuidados y la atención recibida del personal médico	1	0.6	0.6	0.6
	2	0.2	0.2	0.2
	3	0.9	0.9	1
	4	2.1	2.4	1.8
	5	6.8	5.6	7.9
	NO SATISFACTORIO	10.6	9.7	11.5
	6	9.9	9.5	10.4
	7	18.4	20.6	16.2
	8	25.9	27.8	24.1
	TOTAL 6/7/8	54.2	57.9	50.7
	9	15.1	14.4	15.7
	10	18.5	16.3	20.7
	SATISFACTORIO	33.6	30.7	36.4
	N.S.	1.3	1.5	1.1
	N.C.	0.2	0.2	0.3
	#Total cases	2557	1256	1301

Figure 6.5: Uso de subtotales 1

Nótese que en el ejemplo han sido utilizados textos en solo dos de los tres subtotales que se han calculado. Si no se introduce texto, es cuando se usa la palabra TOTAL. Nótese también que por defecto los subtotales aparecen detrás del último valor del grupo, detrás del 5, detrás del 8 y detrás del 10. Existe la posibilidad de determinar mediante una instrucción como deben aparecer. El modificador o parámetro **position** con posibles valores "below", "above", "top" o "bottom" indicarán el lugar donde se deben imprimir. Del mismo modo, se puede forzar a que sea el propio sistema quien determine las etiquetas del subtotal generado. Así el modificador **prefix** puede determinar un prefijo para todas las etiquetas siendo TOTAL el valor por defecto, y también el modificador

`new_label` que permite indicar si la etiqueta se construye usando las etiquetas originales respondiendo a `all` que la usa todas, `range` la primera y la última, `first` la primera del grupo y `last` la última del grupo.

Con todo ello, podríamos modificar nuestro ejemplo a:

```
val_lab(data$P901) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
as.datatable_widget(data %>%
  tab_cols(total(), P31) %>%
  tab_cells(P901) %>%
  tab_subtotal_cells(1:5, 6:8, 9:10, position = "bottom",
    prefix = "SUBT", new_label = "range") %>%
  tab_stat_cpct() %>%
  tab_pivot())
```

		#Total	Sexo de la persona entrevistada	
			Hombre	Mujer
Los cuidados y la atención recibida del personal médico	1	0.6	0.6	0.6
	2	0.2	0.2	0.2
	3	0.9	0.9	1
	4	2.1	2.4	1.8
	5	6.8	5.6	7.9
	6	9.9	9.5	10.4
	7	18.4	20.6	16.2
	8	25.9	27.8	24.1
	9	15.1	14.4	15.7
	10	18.5	16.3	20.7
	N.S.	1.3	1.5	1.1
	N.C.	0.2	0.2	0.3
	SUBT1 - 5	10.6	9.7	11.5
	SUBT6 - 8	54.2	57.9	50.7
	SUBT9 - 10	33.6	30.7	36.4
	#Total cases	2557	1256	1301

Figure 6.6: Uso de subtotales 2

6.5.2 NETS

El uso de `tab_net_rows()` o `tab_net_cols()` o `tab_net_cells()` sustituye por netos (subtotales) a un conjunto de categorías de la variable sobre la que se aplique. Si se introduce un texto se utilizará el mismo, pero si no es así, se añade la palabra TOTAL.

Debes tener en cuenta que si las agrupaciones de categorías que realizas se solapan, también se solaparán los recuentos en el cálculo de netos. Estos netos

pueden ser aplicados a las variables del banco de datos. La terminología de NET suele ser muy aplicada en las variables de tipo múltiple, para agrupar conceptos similares. En nuestro ejemplo por mantener la coherencia con el uso de `subtotal()` lo aplicaremos sin embargo con una variable numérica de valoración.

```
val_lab(data$P901) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
as.datatable_widget(data %>%
  tab_cols(total(), P31) %>%
  tab_cells(P901) %>%
  tab_net_cells(`NO SATISFACTORIO` = 1:5, 6:8, SATISFACTORIO = 9:10) %>%
  tab_stat_cpct() %>%
  tab_pivot())
```

		Sexo de la persona entrevistada		
		#Total	Hombre	Mujer
Los cuidados y la atención recibida del personal médico	NO SATISFACTORIO	10.6	9.7	11.5
	TOTAL 6/7/8	54.2	57.9	50.7
	SATISFACTORIO	33.6	30.7	36.4
	N.S.	1.3	1.5	1.1
	N.C.	0.2	0.2	0.3
#Total cases		2557	1256	1301

Figure 6.7: Uso de nets 1

Nótese que en el ejemplo han sido utilizados textos en solo dos de los tres **nets** que se han calculado. Si no se introduce texto, es cuando se usa la palabra TOTAL. Nótese también que por defecto los subtotales aparecen detrás del último valor del grupo, detrás del 5, detrás del 8 y detrás del 10. Existe la posibilidad de determinar mediante una instrucción como deben aparecer. El modificador o parámetro **position** con posibles valores "below", "above", "top" o "bottom" indicarán el lugar donde se deben imprimir.

Del mismo modo, se puede forzar a que sea el propio sistema quien determine las etiquetas del subtotal generado. Así el modificador **prefix** puede determinar

un prefijo para todas las etiquetas siendo **TOTAL** el valor por defecto, y también el modificador **new_label** que permite indicar si la etiqueta se construye usando las etiquetas originales respondiendo a **all** que la usa todas, **range** la primera y la última, **first** la primera del grupo y **last** la última del grupo. Podríamos modificar nuestro ejemplo a:

```
val_lab(data$P901) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
as.datatable_widget(data %>%
  tab_cols(total(), P31) %>%
  tab_cells(P901) %>%
  tab_net_cells(1:5, 6:8, 9:10, position = "top", prefix = "NET",
    new_label = "range") %>%
  tab_stat_cpct() %>%
  tab_pivot())
```

	#Total	Sexo de la persona entrevistada	
		Hombre	Mujer
Los cuidados y la atención recibida del personal médico	NET1 - 5	10.6	11.5
	NET6 - 8	54.2	50.7
	NET9 - 10	33.6	36.4
	N.S.	1.3	1.1
	N.C.	0.2	0.3
	#Total cases	2557	1301

Figure 6.8: Uso de nets 2

Puedes preguntarte el porqué del **position** si en un NET realmente se eliminan los códigos originales, pero esto no tiene por qué ser así. Tanto el **subtotal** como el **net** tienen la posibilidad de variar este hecho utilizando el modificador **add** con valores **TRUE** o **FALSE** que mantendría o no los códigos originales. No obstante, en nuestro ejemplo como hay valores no agrupados (NS y NC) aquí la posición sí es relevante.

6.6 *Top y Bottom*

Por último, otra utilidad no menos importante que las anterior y no menos utilizada. El cálculo del *top* y el *bottom* de una escala. Para ello, vamos a basarnos en algo que ya hemos visto en las tablas anteriores, la recodificación y el cómo reutilizamos la posibilidad de las variables múltiples.

Planteemos una situación en la que deseamos que la variable P901 se muestre de forma segmentada y conjuntamente cada uno de sus valores. Llamamos TOP a la agrupación en una columna o fila de tabla de aquellas categorías con las valoraciones más altas (por ejemplo 9 y 10) y BOTTOM a las más bajas (por ejemplo 1,2,3,4,5 y 6 o 1:6 como sabemos). Queremos que en la tabla se muestre ambas categorías. ¿Cómo la hacemos? Este es script.

```
val_lab(data$P901) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
as.datatable_widget(data %>%
  tab_cols(total(), P31) %>%
  tab_cells(P901) %>%
  tab_subtotal_cells(Bottom = 1:6, Top = 9:10) %>%
  tab_stat_cpct() %>%
  tab_pivot())
```

	#Total	Sexo de la persona entrevistada	
		Hombre	Mujer
Los cuidados y la atención recibida del personal médico	1	0.6	0.6
	2	0.2	0.2
	3	0.9	1
	4	2.1	1.8
	5	6.8	7.9
	6	9.9	10.4
	Bottom	20.6	21.9
	7	18.4	16.2
	8	25.9	24.1
	9	15.1	15.7
	10	18.5	20.7
	Top	33.6	36.4
	N.S.	1.3	1.1
	N.C.	0.2	0.3
#Total cases	2557	1256	1301

Figure 6.9: Uso de top y bottom

Controlando más algunos aspectos de la publicación de la tabla ...

```

val_lab(data$P901) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
as.datatable_widget(data %>%
  tab_cols(total(), P31) %>%
  tab_cells(P901) %>%
  tab_subtotal_cells(Bottom = 1:6, Top = 9:10, position = "bottom") %>%
  tab_stat_cpct(total_row_position = "below") %>%
  tab_cells(na_if(P901, gt(10))) %>%
  tab_stat_mean() %>%
  tab_pivot())

```

	#Total	Sexo de la persona entrevistada	
		Hombre	Mujer
Los cuidados y la atención recibida del personal médico	1	0.6	0.6
	2	0.2	0.2
	3	0.9	1
	4	2.1	1.8
	5	6.8	7.9
	6	9.9	10.4
	7	18.4	16.2
	8	25.9	24.1
	9	15.1	15.7
	10	18.5	20.7
	N.S.	1.3	1.1
	N.C.	0.2	0.3
	Bottom	20.6	21.9
	Top	33.6	36.4
	#Total cases	2557	1301
	Mean	7.7	7.8

Figure 6.10: Uso de top y bottom arriba

Con subtotales abajo...

```

val_lab(data$P901) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
as.datatable_widget(data %>%
  tab_cols(total(), P31) %>%
  tab_cells(P901) %>%
  tab_subtotal_cells(Bottom = 1:6, Top = 9:10, position = "below") %>%
  tab_stat_cpct(total_row_position = "below") %>%
  tab_cells(na_if(P901, gt(10))) %>%
  tab_stat_mean() %>%
  tab_pivot())

```

		#Total	Sexo de la persona entrevistada	
			Hombre	Mujer
Los cuidados y la atención recibida del personal médico	1	0.6	0.6	0.6
	2	0.2	0.2	0.2
	3	0.9	0.9	1
	4	2.1	2.4	1.8
	5	6.8	5.6	7.9
	6	9.9	9.5	10.4
	Bottom	20.6	19.2	21.9
	7	18.4	20.6	16.2
	8	25.9	27.8	24.1
	9	15.1	14.4	15.7
	10	18.5	16.3	20.7
	Top	33.6	30.7	36.4
	N.S.	1.3	1.5	1.1
	N.C.	0.2	0.2	0.3
	#Total cases	2557	1256	1301
	Mean	7.7	7.7	7.8

Figure 6.11: Uso de top y bottom abajo

O también ...

```
val_lab(data$P901) <- c(`1` = 1, `2` = 2, `3` = 3, `4` = 4,
  `5` = 5, `6` = 6, `7` = 7, `8` = 8, `9` = 9, `10` = 10,
  N.S. = 98, N.C. = 99)
as.datatable_widget(data %>%
  tab_cols(total(), P31) %>%
  tab_cells(P901) %>%
  tab_net_cells(Top = 9:10, Bottom = 1:6) %>%
  tab_stat_cpct(total_row_position = "below") %>%
  tab_cells(na_if(P901, gt(10))) %>%
  tab_stat_mean() %>%
  tab_pivot())
```

	#Total	Sexo de la persona entrevistada	
		Hombre	Mujer
Los cuidados y la atención recibida del personal médico	7	18.4	16.2
	8	25.9	24.1
	Top	33.6	36.4
	Bottom	20.6	21.9
	N.S.	1.3	1.1
	N.C.	0.2	0.3
#Total cases	2557	1256	1301
Mean	7.7	7.7	7.8

Figure 6.12: Uso de top y bottom sin los valores agrupados

Así pues en esta última tabla, hemos combinado algunas de las funcionalidades especiales y avanzadas de `expss` de forma conjunta y trabajando en la misma dirección.

6.6.1 Gráficos base

Dejamos además aquí esta `pildora adictiva` adelantándonos a la sección ???. Vamos a crear el primer gráfico en este libro. Vamos a repetir la tabla anterior, pero la guardamos en un objeto llamado `tab`. Limpiamos además aquello que no nos interesa, lo convertimos y lo publicamos en su formato bruto o estándar, es decir como *dataframe*.

```
tab <- data %>%
  tab_cols(`|` = unvr(P31)) %>%
  tab_cells(`|` = unvr(P901)) %>%
  tab_net_cells(Detractores = 1:6, Neutrales = 7:8, Promotores = 9:10) %>%
  tab_stat_cpct(total_row_position = "none") %>%
  tab_pivot()
tab <- as.data.frame(tab)
tab
```

```
##   row_labels   Hombre   Mujer
## 1 Detractores 19.1878981 21.9062260
## 2  Neutrales 48.4076433 40.3535742
## 3  Promotores 30.7324841 36.3566487
```



```
## 4      N.S.  1.5127389  1.0760953
## 5      N.C.  0.1592357  0.3074558
```

Fíjate como hemos limpiado la tabla. Hemos usado un recurso que para gráficos será muy válido, el uso de `unvr()` que elimina del uso de la variable los textos extra. Del mismo modo con el `"|"` hemos anulado cualquier tipo de texto suplementario que se pudiera añadir. Cuando hacemos una tabla y la guardamos, es un objeto `etable` que ya describimos anteriormente. Como la mayoría de los paquetes gráficos usan `_dataframe_`, lo transformamos a eso.

Una vez hecho esto, hacemos el gráfico. Podemos elegir diferentes sistemas de gráficos, pero los más habituales e interactivos son `highcharter` (paquete escrito sobre Highcharts) y/o `plotly`. Como paquete estático, `ggplot` sería la mejor solución.

```
suppressMessages(library(highcharter, quietly = TRUE))
highchart() %>%
  hc_xAxis(categories = tab$row_labels) %>%
  hc_add_series(data = tab, type = "column", hcaes(x = row_labels,
    y = Hombre), name = "hombre") %>%
  hc_add_series(data = tab, type = "column", hcaes(x = row_labels,
    y = Mujer), name = "mujer")
```

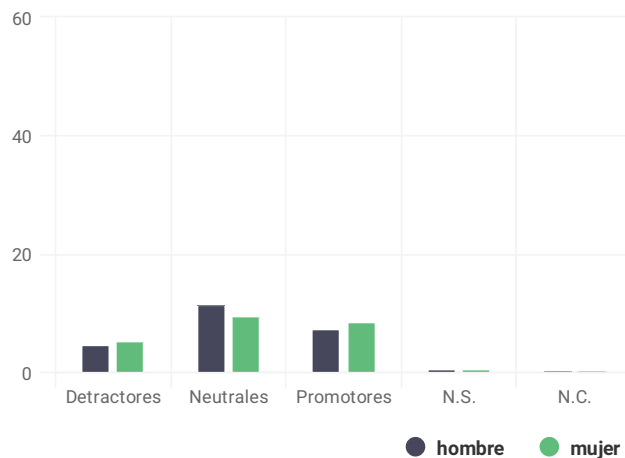


Figure 6.13: Uso de gráfico highcharter 2

Alternativamente, si queremos utilizar el paquete `plotly` ...

```
Grupo <- factor(tab$row_labels, levels = tab$row_labels)
plot_ly(tab, x = ~Grupo, y = ~Hombre, type = "bar", name = "Hombre") %>%
  add_trace(y = ~Mujer, name = "Mujer") %>%
  layout(yaxis = list(title = "frecuencia"))
```

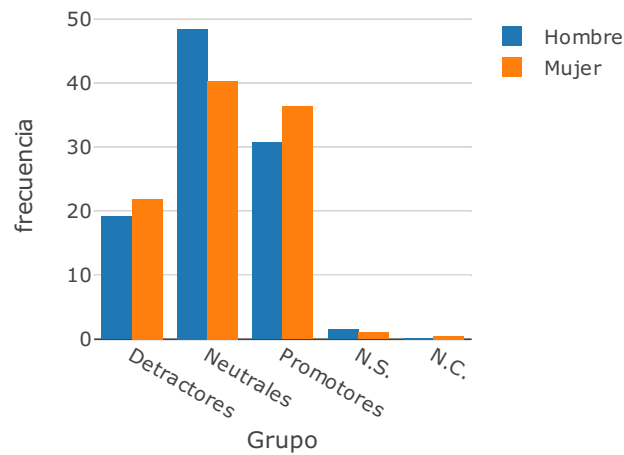


Figure 6.14: Uso de gráfico con `plotly`

Y esto es todo en cuanto a tablas básicas. La siguiente sección (7), con un carácter más estadístico, se introduce en las pruebas de significación que se usan en las tablas de contingencia. Si no te resultan de interés o si ya estás con ganas de gráficos, puedes saltar a la sección (??) y sumergirte en el mundo gráfico.

Chapter 7

Pruebas inferenciales

Cuando trabajamos con tablas de contingencia es muy frecuente que sintamos la necesidad de tener que inferir acerca de la dependencia de las categorías analizadas o de las diferencias entre los grupos analizados. Siempre que nuestras variables cumplan con los requisitos que para ellas cada prueba establece (normalidad, homocedasticidad, linealidad y en algunos casos independencia), podremos aplicar las pruebas inferenciales típicas con tablas de contingencia en la investigación básica:

- Chi2 en su variantes de tabla y celda (??,);
- Pruebas z de contraste proporciones (??,);
- Prueba t de contraste de medias (??,).

Para todas ellas `expss` nos da la oportunidad de hacer los cálculos desde el propio script de realización de la tabla y/o desde una instrucción posterior a la realización de la tabla. Pasemos por ello a explicar, no tanto el cometido de estas pruebas, sino el como llevarlas adelante.

7.1 Prueba de dependencia

El contraste Chi2 de Pearson es una prueba estadística no paramétrica, que compara las frecuencias realmente obtenidas con las frecuencias esperadas que son las que corresponderían a cada celda o casilla de la tabla si su valor se ajustase a cualquier norma teórica previamente adoptada; en nuestro caso, una distribución proporcional de frecuencias normales ?. En definitiva, “*se está calculando un índice acerca de la distancia entre lo real y lo esperado*” ?.

El valor numérico de esta prueba se obtiene como:

$$\chi^2 = \sum_{i=1} \frac{(fo - fe)^2}{fe}$$

- fo, serán las frecuencias observadas en el experimento o muestra
- fe, serán las frecuencias esperadas teóricamente

Las frecuencias esperadas se calculan con ...

$$fe = \frac{N_{columna} * N_{fila}}{N_{total}}$$

- fo, serán las frecuencias observadas en el experimento o muestra
- fe, serán las frecuencias esperadas teóricamente
- N, es el número de efectivos muestrales

Esta prueba se suele utilizar (entre muchas otras posibilidades) para contrastar la hipótesis nula que los resultados obtenidos de una muestra no son significativos con relación a la población total, o bien como prueba de dependencia para comprobar la existencia o no de asociación entre las variables. En este caso, la prueba indica la existencia de asociación pero no la cuantifica ?.

7.1.1 De una tabla

La prueba Chi2 puede hacerse a nivel de tabla, lo que muestra la relación de dependencia entre las categorías ?. Hagamos una primera aproximación con dos tablas de contingencia muy sencillas, pero que nos mostrarán como se indica que la relación de dependencia existe o no existe. La función `tab_last_sig_cases` realiza la prueba base de R denominada `chisq.test`.

Nótese el uso de “|”=`unvr()` para utilizar la variable sin que se publiquen los texto extra de la misma.

```
as.datatable_widget(data %>%
  tab_cols(total(), `|` = unvr(P31)) %>%
  tab_cells(`|` = unvr(P2)) %>%
  tab_stat_cases() %>%
  tab_last_sig_cases() %>%
  tab_pivot())
```

	#Total	Hombre	Mujer
En general, el sistema sanitario funciona bastante bien	538.0	277.0	261.0
El sistema sanitario funciona bien, aunque son necesarios al	1247.0	620.0	627.0
El sistema sanitario necesita cambios fundamentales, aunque	637.0	288.0	349.0
Nuestro sistema sanitario está tan mal que necesitaríamos re	120.0	61.0	59.0
N.S.	9.0	6.0	3.0
N.C.	6.0	4.0	2.0
#Chi-squared p-value	(warn.)		
#Total cases	2557.0	1256.0	1301.0

Figure 7.1: Prueba χ^2 en tabla

```
as.datatable_widget(data %>%
  tab_cols(total(), `|` = unvr(P31)) %>%
  tab_cells(`|` = unvr(P33)) %>%
  tab_stat_cases() %>%
  tab_last_sig_cases() %>%
  tab_pivot())
```

	#Total	Hombre	Mujer
Casado/a	1388.0	677.0	711.0
Soltero/a	817.0	455.0	362.0
Viado/a	190.0	41.0	149.0
Separado/a	57.0	24.0	33.0
Divorciado/a	97.0	55.0	42.0
N.C.	8.0	4.0	4.0
#Chi-squared p-value	<0.05 (warn.)		
#Total cases	2557.0	1256.0	1301.0

Figure 7.2: Prueba χ^2 en tabla

En la primera tabla se muestra la relación entre la variable P31 (sexo) y la P2 (valoración del sistema sanitario). Nótese que en la tabla se ha usado una línea tras el cálculo de los casos con la función `tab_last_sig_cases()` que indica que se debe realizar la prueba Chi2 a la relación. Esta línea provoca que en la tabla surja una nueva fila sobre el **#Total** con el texto **#Chi-squared p-value** que indica que se realiza la prueba al 5% (0,05). Si el resultado es el rechazo de la hipótesis nula de independencia se muestra un **<0,05 (warn.)**, pero si no se puede rechazar la hipótesis nula de independencia sale sólo **(warn.)** En la tabla no se publica el resultado de la prueba, pero podemos hacerlo siguiendo el formato estándar.

```
table(data$P2, data$P31)
```

```
##
##
##      En general, el sistema sanitario funciona bastante bien      Hombre
##      El sistema sanitario funciona bien, aunque son necesarios al 620
##      El sistema sanitario necesita cambios fundamentales, aunque 288
##      Nuestro sistema sanitario está tan mal que necesitaríamos re 61
##      N.S.                                                         6
##      N.C.                                                         4
##
##
##      Mujer
##      En general, el sistema sanitario funciona bastante bien      261
##      El sistema sanitario funciona bien, aunque son necesarios al 627
##      El sistema sanitario necesita cambios fundamentales, aunque 349
##      Nuestro sistema sanitario está tan mal que necesitaríamos re 59
##      N.S.                                                         3
##      N.C.                                                         2
```

```
chisq.test(table(data$P2, data$P31))
```

```
##
##      Pearson's Chi-squared test
##
##      data:  table(data$P2, data$P31)
##      X-squared = 7.2669, df = 5, p-value = 0.2015
```

```
table(data$P33, data$P31)
```

```
##
##      Hombre Mujer
##      Casado/a    677  711
##      Soltero/a   455  362
```

```
## Viudo/a      41  149
## Separado/a   24   33
## Divorciado/a 55   42
## N.C.         4    4
```

```
chisq.test(table(data$P33, data$P31))
```

```
##
## Pearson's Chi-squared test
##
## data:  table(data$P33, data$P31)
## X-squared = 75.203, df = 5, p-value = 8.437e-15
```

Donde se puede observar que para la primera relación, no se puede rechazar la hipótesis de independencia pues el valor de significación es $p\text{-value} > 0,05$ (0.2015); para la segunda relación, sí podemos rechazar la hipótesis nula de independencia, puesto que $p\text{-value} < 0,05$ (por tanto, existe dependencia).

7.1.2 De una celda de una tabla

Particularmente de interés en investigación de mercados (de hecho solo está documentado su uso, y poco, en este ámbito) es la prueba Chi2 de celda. A diferencia de la anterior, en este caso se realiza la prueba para cada celda de la tabla en particular. La lógica de la misma sería comparar un valor de la tabla (una celda), con el resto de su fila, el resto de su columna, y el resto de la muestra. De este forma, indicamos que valores son significativos en la tabla, aquellos que cabría contemplar con un interés especial.

Para obtener la tabla y la subsiguiente prueba se utilizará una nueva función denominada `tab_last_sig_cell_chisq()` sobre la misma estructura ya conocida de tabla. Nótese que en este caso, para la prueba se requiere utilizar los porcentajes en lugar de los casos, para que el cálculo sea el oportuno. Chi2 es una prueba muy sensible al tamaño de la muestra.

```
as.datatable_widget(data %>%
  tab_cols(total(), `|` = unvr(P31)) %>%
  tab_cells(`|` = unvr(P2)) %>%
  tab_stat_cpct() %>%
  tab_last_sig_cell_chisq() %>%
  tab_pivot())
```

	#Total	Hombre	Mujer
En general, el sistema sanitario funciona bastante bien	21.0	22.1	20.1
El sistema sanitario funciona bien, aunque son necesarios al	48.8	49.4	48.2
El sistema sanitario necesita cambios fundamentales, aunque	24.9	22.9 <	26.8 >
Nuestro sistema sanitario está tan mal que necesitaríamos re	4.7	4.9	4.5
N.S.	0.4	0.5	0.2
N.C.	0.2	0.3	0.2
#Total cases	2557	1256	1301

Figure 7.3: Prueba χ^2 de celda

```
as.datatable_widget(data %>%
  tab_cols(total(), `|` = unvr(P31)) %>%
  tab_cells(`|` = unvr(P33)) %>%
  tab_stat_cpct() %>%
  tab_last_sig_cell_chisq() %>%
  tab_pivot())
```

	#Total	Hombre	Mujer
Casado/a	54.3	53.9	54.7
Soltero/a	32.0	36.2 >	27.8 <
Viudo/a	7.4	3.3 <	11.5 >
Separado/a	2.2	1.9	2.5
Divorciado/a	3.8	4.4	3.2
N.C.	0.3	0.3	0.3
#Total cases	2557	1256	1301

Figure 7.4: Prueba χ^2 de celda

La salida es muy clara. Con los símbolos mayor y menor, se marcan aquellas celdas que son significativamente mayores ($>$) o menores ($<$) que lo esperado y por tanto son las que direccionan las relaciones de dependencia que en la tabla se producen.

7.2 Pruebas de diferencias

Un conjunto diferentes de pruebas son aquellas cuya hipótesis de partida se basa en determinar si existen diferencias entre los porcentajes (prueba z) o las medias (prueba t) de dos grupos independientes en la muestra extraídos de la misma población. Desarrollamos ambas pruebas en las líneas siguientes.

7.2.1 Porcentajes (prueba z)

Asumiendo las hipótesis necesarias para poder trabajar con estadística paramétrica (normalidad, homoscedasticidad, linealidad y en algunos casos independencia), la función `tab_last_sig_cpct` realiza z-test entre columnas de porcentajes derivadas de la aplicación de `tab_stat_cpct`. Los resultados son calculados con la misma fórmula que con la función base de R `prop.test` y sin la corrección de continuidad.

Obsérvese la diferencia de concepto; mientras que la prueba Chi2 de celda realiza la prueba comparando con el marginal total, la **prueba z** realiza esa comparación entre los grupos formados por las columnas, a los que se suele llamar perfiles. De esta forma considera la independencia de los grupos muestrales entre sí.

Para utilizar esta funcionalidad el script sería el siguiente:

```
as.datatable_widget(data %>%
  tab_cols(total(), SEXO = unvr(P31)) %>%
  tab_cells(`|` = unvr(P2)) %>%
  tab_stat_cpct() %>%
  tab_last_sig_cpct() %>%
  tab_pivot())
```

	#Total	SEXO	
		Hombre	Mujer
		A	B
En general, el sistema sanitario funciona bastante bien	21.0	22.1	20.1
El sistema sanitario funciona bien, aunque son necesarios al	48.8	49.4	48.2
El sistema sanitario necesita cambios fundamentales, aunque	24.9	22.9	26.8 A
Nuestro sistema sanitario está tan mal que necesitaríamos re	4.7	4.9	4.5
N.S.	0.4	0.5	0.2
N.C.	0.2	0.3	0.2
#Total cases	2557	1256	1301

Figure 7.5: Prueba Z en tabla

```
as.datatable_widget(data %>%
  tab_cols(total(), SEXO = unvr(P31)) %>%
  tab_cells(`|` = unvr(P33)) %>%
  tab_stat_cpct() %>%
  tab_last_sig_cpct() %>%
  tab_pivot())
```

	#Total	SEXO	
		Hombre	Mujer
		A	B
Casado/a	54.3	53.9	54.7
Soltero/a	32.0	36.2 B	27.8
Viudo/a	7.4	3.3	11.5 A
Separado/a	2.2	1.9	2.5
Divorciado/a	3.8	4.4	3.2
N.C.	0.3	0.3	0.3
#Total cases	2557	1256	1301

Figure 7.6: Prueba Z en tabla

En nuestro caso, los resultados son muy semejantes a los vistos con Chi2 de celda, porque la variable elegida para las columnas es dicotómica, es decir, con sólo dos opciones de respuesta, exhaustivas y mutuamente excluyentes. No sería así si la variable de columnas presentara más de 2 perfiles.

La lectura de esta prueba es la siguiente. El porcentaje de casos en el grupo B (mujeres) de la tabla 1, es significativamente más elevado que el de hombres, determinándose esta diferencia con una significación del 5%. En el caso de la tabla 2, el porcentaje de hombres solteros es significativamente diferente del porcentaje de mujeres solteras. Del mismo modo y a la inversa el porcentaje de mujeres viudas entrevistadas en la muestra es significativamente mayor que el de hombres.

Por tanto, creemos que queda claro el funcionamiento de la prueba. Se etiquetan las columnas y se muestra la letra de la columna con la que se presentan diferencias positivas junto al valor porcentual. La prueba se realiza para cada celda, pero siempre comparando con las celdas que tiene a su derecha o izquierda en la misma fila (no con el total).

7.2.2 Medias (prueba t)

Al igual que en el apartado anterior el objetivo es determinar si existen o no diferencias entre los grupos que se están testando, teniendo como hipótesis nula que las medias de los grupos son iguales. En nuestro ejemplo, hemos tomado la de auto clasificación ideológica (recodificando las posiciones de 1 a 10, izquierda a derecha respectivamente) creando grupos de izquierda, centro y derecha. Sobre esta tabla que calcula las medias, se aplica el estadístico `tab_stat_mean_sd_n()` que contiene todos los datos requeridos para el cálculo del valor t y se le indica que requerimos el test con `tab_last_sig_means()`. Se asume que los grupos son independientes, que existe normalidad y que las varianzas de los grupos son iguales.

```
as.datatable_widget(data %>%
  tab_cols(total(), P29 = recode(P29, Izquierda = 1:4 ~
    1, Centro = 5:6 ~ 2, Derecha = 7:10 ~ 3, TRUE ~ NA)) %>%
  tab_cells(P3 = na_if(P3, gt(10))) %>%
  tab_stat_mean_sd_n() %>%
  tab_last_sig_means() %>%
  tab_pivot())
```

	#Total	P29		
		Izquierda	Centro	Derecha
		A	B	C
Escala de satisfacción (1-10) con el funcionamiento del sistema sanitario español	Mean	6.8	6.7	6.8
	Std. dev.	1.9	1.9	1.8
	Unw. valid N	2542.0	770.0	750.0
				348.0

Figure 7.7: Prueba t en tabla

Se puede observar que la salida es igual a la de la prueba Z. Se rotulan las columnas con las letras A, B ... y las que sean necesarias, y posteriormente se muestra (por defecto) la letra de la columna con la que la media de la columna en la que se ubica la media presenta diferencias positivas (es mayor). Podemos por tanto observar, que en la población de la que se ha extraído la muestra, se puede afirmar que la media de satisfacción con el funcionamiento del sistema sanitario español es más alta en los individuos cuya auto clasificación ideológica es del grupo de derecha (C), que en la izquierda (A) y en el centro (B). No entramos a valorar si la distribución de grupos es la correcta o no, en cuanto al significado general. Se ha hecho una distribución acorde al significado de los números en sí mismos.

Existen ocasiones en las que esta prueba, se requiere publicar para un conjunto de ítems que forman parte de una misma batería. En estos casos, no es tan interesante publicar las desviaciones y las bases, por lo que podemos formular de esta forma el script.

```
as.datatable_widget(data %>%
  tab_cols(total(), P29 = recode(P29, Izquierda = 1:4 ~
    1, Centro = 5:6 ~ 2, Derecha = 7:10 ~ 3, TRUE ~ NA)) %>%
  tab_cells(P901 = na_if(P901, gt(10))) %>%
  tab_stat_mean_sd_n() %>%
  tab_last_sig_means(keep = "means") %>%
  tab_cells(P902 = na_if(P902, gt(10))) %>%
  tab_stat_mean_sd_n() %>%
  tab_last_sig_means(keep = "means") %>%
  tab_cells(P903 = na_if(P903, gt(10))) %>%
```

```

tab_stat_mean_sd_n() %>%
tab_last_sig_means(keep = "means") %>%
tab_cells(P904 = na_if(P904, gt(10))) %>%
tab_stat_mean_sd_n() %>%
tab_last_sig_means(keep = "means") %>%
tab_cells(P905 = na_if(P905, gt(10))) %>%
tab_stat_mean_sd_n() %>%
tab_last_sig_means(keep = "means") %>%
tab_cells(P906 = na_if(P906, gt(10))) %>%
tab_stat_mean_sd_n() %>%
tab_last_sig_means(keep = "means") %>%
tab_cells(P907 = na_if(P907, gt(10))) %>%
tab_stat_mean_sd_n() %>%
tab_last_sig_means(keep = "means") %>%
tab_pivot()

```

		#Total	P29		
			Inquierda	Centro	Derecha
			A	B	C
Los cuidados y la atención recibida del personal médico	Mean	7.7	7.7	7.8	7.9
Los cuidados y la atención recibida del personal de enfermería	Mean	7.8	7.8	7.7	7.9
La confianza y seguridad que transmite el personal médico	Mean	7.8	7.7	7.8	8.0 A
La confianza y seguridad que transmite el personal de enfermería	Mean	7.8	7.7	7.7	8.0 AB
El tiempo dedicado por el médico o la médica a cada enfermo o enferma	Mean	7.1	6.9	7.0	7.2
El conocimiento del historial y seguimiento de los problemas de salud de cada usuario o usuaria	Mean	7.5	7.4	7.4	7.7 AB
La información recibida sobre su problema de salud	Mean	7.5	7.4	7.5	7.8 AB

Figure 7.8: Prueba t con sólo medias

Se puede observar que la instrucción `keep="means"` lo que ha conseguido es eliminar la publicación de la desviación y la media del cuadro presentado. De este modo el resultado es más compacto y da una visión general de la batería de ítems

7.3 Parámetros posibles en las pruebas de significación

De manera conjunta exponemos aquí diferentes parámetros que modifican el comportamiento por defecto de las cuatro pruebas anteriormente vistas. Algunos son de uso en todas ellas y otros específicos de alguna de las pruebas.

- **sig_level**, numérico; nivel de significación, por defecto es igual a 0.05.
- **min_base**, numérico; el test de significación se realizará si ambas columnas tienen bases mayores o iguales al valor determinado que por defecto es 2.
- **delta_cpct**, numérico; delta mínimo entre el porcentaje para el que marcamos diferencias significativas (en puntos porcentuales); de forma predefinida, es igual a cero. Tenga en cuenta que, por ejemplo, para una diferencia mínima de 5 por ciento de puntos, **delta_cpct** debe ser igual a 5, no 0.05.
- **delta_means**, numérico; delta mínimo entre medias para las que marcamos diferencias significativas: por defecto es igual a cero.
- **correct**, lógico (TRUE o FALSE), indica si aplicar corrección de continuidad al calcular el estadístico Chi2 de prueba para tablas de 2 por 2. Solo para **significance_cases** y **significance_cell_chisq**. Para más detalles ver **chisq.test**. TRUE por defecto.
- **compare_type** tipo de comparación por columnas. Por defecto, es subtabla (variable por variable). otras posibilidades son **"first_column"**, **"adjusted_first_column"** y **"previous_column"**; podemos realizar varios test simultáneamente.
- **bonferroni** lógico; FALSE por defecto; uso del ajuste de Bonferroni por cada fila.
- **subtable_marks**, carácter; una de las siguientes opciones: **"greater"**, **"both"** or **"less"**; por defecto se marcan sólo valores cuya significación sea mayor (**"greater"**) que alguna otra columna. Para **significance_cell_chisq** por defecto es **"both"**. podemos modificar este comportamiento usando las otras alternativas.
- **inequality_sign** logical. FALSE if **subtable_marks** is "less" or "greater". Should we show > or < before significance marks of subtable comparisons.
- **sig_labels** character vector labels for marking differences between columns of subtable.
- **sig_labels_previous_column** a character vector with two elements. Labels for marking a difference with the previous column. First mark means 'lower' (by default it is v) and the second means greater (^).
- **sig_labels_first_column** a character vector with two elements. Labels for marking a difference with the first column of the table. First mark means 'lower' (by default it is -) and the second means 'greater' (+).
- **sig_labels_chisq** a character vector with two labels for marking a difference with row margin of the table. First mark means 'lower' (by

default it is $<$) and the second means 'greater' ($>$). Only for significance_cell_chisq.

- **keep**, carácter. Una o más de las siguientes "percent", "cases", "means", "bases", "sd" o "none". Este argumento determina qué estadísticos permanecerán en la tabla después del marcado de significación.
- **row_margin**, carácter. Uno de los valores "auto" (predeterminado), "sum_row" o "first_column". Si es "auto", tratamos de encontrar la columna total en la subtabla por **total_column_marker**. Si la búsqueda falla, usamos la suma de cada fila como total de filas. Con la opción "sum_row" siempre sumamos cada fila para obtener margen. Tenga en cuenta que en este caso el resultado de las variables de respuesta múltiple en la cabecera puede ser incorrecta. Con la opción "first_column" usamos la tabla primera columna como margen de fila para todas las subtablas. En este caso, el resultado de las subtablas con bases incompletas puede ser incorrecto. Solo para **significance_cell_chisq**.
- **total_marker**, carácter. Total de fila marcado en la tabla. " # " por defecto.
- **total_row**, entero/carácter. En el caso de varios totales por subtabla, es un número o nombre de fila total para el cálculo de significación.
- **digits**, un número entero que indica cuántos dígitos después del separador decimal se mostrarán en la tabla final.
- **na_as_zero**, lógico; FALSE por defecto. ¿Deberíamos tratar a NA como cero casos?
- **var_equal**, lógico; variable que indica si se deben tratar las dos varianzas como iguales. Para más detalles ver t.test.
- **mode**, carácter; "replace" (default) o "append". En el primer caso, el resultado anterior en la secuencia del cálculo de la tabla se reemplazará con el resultado de la prueba de significación. En el segundo caso, el resultado de la prueba de significación se agregará a la secuencia del cálculo de la tabla.
- **label**, carácter; etiqueta para la estadística en tab_*. Ignorado si el modo es igual a **replace**.
- **total_column_marker**, carácter; marca para la columna de totales en las subtablas. "#" por defecto.
- **x** table (class etable): result of cro_cpct with proportions and bases for significance_cpct, result of cro_mean_sd_n with means, standard deviations and valid N for significance_means, and result of cro_cases with counts and bases for significance_cases.
- **cases_matrix**, matriz numérica con recuentos de tamaño filas*columnas.
- **row_base**, vector de números con las bases de fila.
- **col_base**, vector de números con las bases de columna.
- **total_base**, número con la base total.

7.3.1 Algunos ejemplos de uso de los parámetros

Cambio del nivel de significación de la prueba y eliminación de las filas con las frecuencias, entre otros...

```
as.datatable_widget(data %>%
  tab_cols(total(), `|` = unvr(P31)) %>%
  tab_cells(`|` = unvr(P33)) %>%
  tab_stat_cases() %>%
  tab_last_sig_cases(sig_level = 0.01, correct = TRUE,
    keep = "bases", mode = "replace", label = "***") %>%
  tab_pivot())
```

```
\begin{figure}[H]
```

	#Total	Hombre	Mujer
#Chi-squared p-value	<0.01 (wam)		
#Total cases	2557.0	1256.0	1301.0

```
{
```

```
}
```

```
\caption{Prueba Chi2 con significación al 99%} \end{figure}
```

7.4 Conclusión al uso de expss

Hasta aquí llegamos. Hemos presentado de forma muy breve y simplificada como podemos aprovechar toda la potencia de **expss** en nuestros scripts. Lo importante es practicar y practicar. No dejes de acudir a las viñetas de ayuda de Gregory Demin acerca de como usar el paquete y como generar nuevas tablas. Nosotros tan sólo hemos sentado las bases. Combinando las tablas con lenguaje R se puede llegar a conseguir casi todo.

- manual PDF de *EXPSS*
- material de ayuda, ejemplos
- uso de etiquetas en R

Chapter 8

Estilos y formatos de tabla

En cuanto a operativa de trabajo, esta es nuestra última sección, puesto que la última es complementaria. En esta sección aprenderemos a combinar gráficos y tablas con visualizaciones especiales, de la forma más diversa.

8.1 Introducción

Los paquetes que hemos utilizado en nuestro trabajo con las tablas ya han sido referido en multitud de ocasiones. **expss** nos permite además de obtener el cuadro preciso, operar con las filas y columnas de la tabla, puesto que el objeto creado es como un *dataframe* especial, ya que su clase de objeto es **etable**. Por tanto, vamos a tener la capacidad de poder operar con las columnas (variable) así como también con las filas (registros de ese *dataframe*).

Existen multitud de paquetes con los que poder interactuar para poder obtener los resultados deseados. Los paquetes **DT** o **_datatable_**, **formattable**, y también **kable** y **kableExtra** que son nativos para utilizar junto con **rmarkdown**, lo que conforma un amplio espectro de soluciones. Debemos buscar aquella que responda a nuestras expectativas y que haga nos sintamos más cómodos. Nuestra elección será **kableExtra**.

Por el lado de las operaciones en la tabla, comenzaremos sentando las bases de como se realizan las operaciones, para luego trabajar la salida aportando estilos y formatos condicionales a las mismas.

8.2 Operaciones con tablas

Como hemos indicado en la presentación anterior, nuestro primer paso será la realización de operaciones muy simples con las tablas, que seguro en nuestro

trabajo estamos acostumbrados a hacer. también vamos a adentrarnos un poco en la creación de los elementos con los que vamos a trabajar. Ahora ya no trabajaremos sobre lo que ha sido nuestro fichero base, sino que iremos aportando nuestros propios ejemplos que se adecuen a lo que deseamos hacer.

Imaginemos que disponemos datos sobre las cifras de ventas anuales de cinco empresas muy conocidas: Apple, Amazon, Microsoft, Google y Facebook. Estas empresas ofrecen información sobre Ingresos , Beneficio Operativo y Beneficio Neto de cuatro trimestre de 2018 y primer trimestre de 2019 en millones de dólares. Con estos datos creamos un *dataframe*.

```
data <- data.frame(emp = c("Apple", "Apple", "Microsoft",
  "Microsoft", "Amazon", "Amazon", "Google", "Google",
  "Facebook", "Facebook"), ing = c(84310, 91819, 32471,
  36906, 72400, 87400, 39276, 46075, 16914, 21082), bfo = c(23346,
  25569, 10258, 13891, 3786, 3879, 8221, 9266, 7820, 8858),
  bfn = c(19965, 22236, 8420, 11649, 3027, 3268, 8948,
  10671, 6882, 7349), per = c("IV-2018", "I-2019",
  "IV-2018", "I-2019", "IV-2018", "I-2019", "IV-2018",
  "I-2019", "IV-2018", "I-2019"))
data
```

```
##      emp  ing  bfo  bfn  per
## 1   Apple 84310 23346 19965 IV-2018
## 2   Apple 91819 25569 22236 I-2019
## 3 Microsoft 32471 10258  8420 IV-2018
## 4 Microsoft 36906 13891 11649 I-2019
## 5   Amazon 72400  3786  3027 IV-2018
## 6   Amazon 87400  3879  3268 I-2019
## 7   Google 39276  8221  8948 IV-2018
## 8   Google 46075  9266 10671 I-2019
## 9 Facebook 16914  7820  6882 IV-2018
## 10 Facebook 21082  8858  7349 I-2019
```

Ahora vamos a obtener la tabla con la que deseamos trabajar, que sería comparar los ingresos de las cinco compañías en los dos períodos. Para ello creamos una tabla de la siguiente forma.

```
as.datatable_widget(data %>%
  tab_cols(per) %>%
  tab_rows(emp) %>%
  tab_cells(ing) %>%
  tab_stat_sum() %>%
  tab_pivot())
```

				per	
				I-2019	IV-2018
emp	Amazon	ing	Sum	87400	72400
	Apple	ing	Sum	91819	84310
	Facebook	ing	Sum	21082	16914
	Google	ing	Sum	46075	39276
	Microsoft	ing	Sum	36906	32471

Figure 8.1: Tabla con formato estándar de ‘expss’

Este sería el formato estándar de salida, pero como ya hemos visto en anteriores secciones, podemos adaptar esa salida a nuestras necesidades. Nótese que aun no siendo necesario, vamos a guardar la tabla en un output llamado **tab01** que luego vamos a publicar. Utilizamos la función `class()` para mostrarte que tras hacer la tabla, el output almacenado es un *etable dataframe*.

```
tab01 <- data %>%
  tab_cols(`|` = unvr(per)) %>%
  tab_rows(`|` = unvr(emp)) %>%
  tab_cells(`|` = unvr(ing)) %>%
  tab_stat_sum(label = "|") %>%
  tab_pivot()
class(tab01)
```

```
## [1] "etable"      "data.frame"
```

```
as.datatable_widget(tab01)
```

	I-2019	IV-2018
Amazon	87400	72400
Apple	91819	84310
Facebook	21082	16914
Google	46075	39276
Microsoft	36906	32471

Figure 8.2: Tabla con formato adaptado de ‘expss’

Podemos ver que esta tabla ya tiene un formato más adecuado a nuestra necesidad. Hemos quitado aquellos textos que en esta ocasión eran innecesarios.

Si convertimos (aunque ya lo es) y mostramos esta tabla como *dataframe* el resultado sería éste.

```
tab01 <- as.data.frame(tab01)
class(tab01)
```

```
## [1] "data.frame"
```

```
tab01
```

```
##   row_labels I-2019 IV-2018
## 1   Amazon  87400   72400
## 2   Apple   91819   84310
## 3 Facebook  21082   16914
## 4   Google  46075   39276
## 5 Microsoft 36906   32471
```

Nótese que ahora ya el *output* no es un objeto de tipo *etable*, solo es *dataframe*. Ya estamos en condiciones de poder operar. Calculemos la diferencia entre los dos trimestres.

```
tab01$dif <- tab01[, 2] - tab01[, 3]
tab01
```

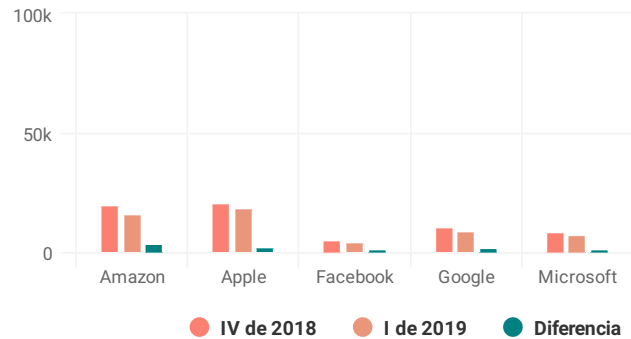
```
##   row_labels I-2019 IV-2018   dif
## 1   Amazon  87400   72400 15000
## 2   Apple   91819   84310  7509
## 3 Facebook  21082   16914  4168
## 4   Google  46075   39276  6799
## 5 Microsoft 36906   32471  4435
```

Esta información puede ser presentada en un gráfico tal como veíamos en la sección anterior. Aprovechemos también para cambiar el nombre de las columnas ...

```
colnames(tab01) <- c("empresa", "2019 (1º)", "2018 (4º)",
  "dif")
highchart() %>%
  hc_chart(type = "column") %>%
  hc_title(text = "Cifra de negocio de las 5 mayores tecnológicas") %>%
  hc_xAxis(categories = tab01[, 1]) %>%
  hc_yAxis(min = 0, max = 1e+05) %>%
  hc_add_series(data = tab01[, 2], name = "IV de 2018",
    dataLabels = list(enabled = TRUE), color = "#FA8072") %>%
  hc_add_series(data = tab01[, 3], name = "I de 2019",
    dataLabels = list(enabled = TRUE), color = "#E9967A") %>%
  hc_add_series(data = tab01[, 4], name = "Diferencia",
    dataLabels = list(enabled = TRUE), color = "teal")
```

\begin{figure}[H]

Cifra de negocio de las 5 mayores tecnológicas



```
{
```

```
}
```

```
\caption{Gráfico del dataframe} \end{figure}
```

Añadamos más información a la tabla. Vamos a calcular el porcentaje de incremento. Habrían muchas formas de hacerlo, pero vamos a tratar de hacerlo de la forma más simple.

```
tab01$difpct <- round(((tab01[, 2]/tab01[, 3]) - 1) * 100,
1) #cociente entre valores y paso a porcentaje con redondeo a un decimal
tab01
```

```
##      empresa 2019 (1º) 2018 (4º)   dif difpct
## 1   Amazon      87400    72400 15000    20.7
## 2    Apple      91819    84310  7509     8.9
## 3 Facebook      21082    16914  4168    24.6
## 4   Google      46075    39276  6799    17.3
## 5 Microsoft     36906    32471  4435    13.7
```

Esto nos va a permitir probar algo que no vimos en la sección anterior, presentar en función de otro eje Y la información. Se puede observar que en el script abajo referido, estamos creando en la función `hc_yAxis()` una lista de dos ejes a los que referenciar los datos, mientras que las cifras absolutas se miran con el eje Y de la izquierda, el dato relativo se mira con el eje Y a la derecha del gráfico. El resto de opciones ya fueron analizadas

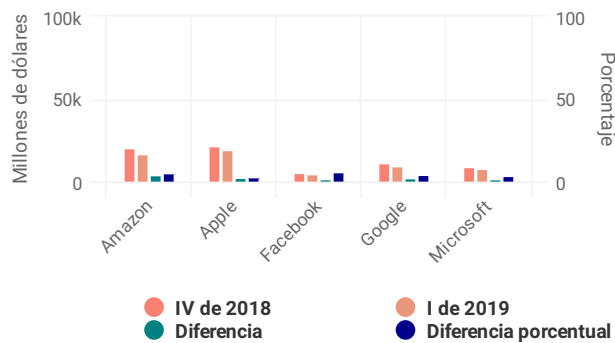

```

highchart() %>%
  hc_chart(type = "column") %>%
  hc_title(text = "Cifra de negocio de las 5 mayores tecnológicas") %>%
  hc_xAxis(categories = tab01[, 1]) %>%
  hc_yAxis_multiples(list(title = list(text = "Millones de dólares"),
    min = 0, max = 1e+05), list(title = list(text = "Porcentaje"),
    min = 0, max = 100, opposite = TRUE)) %>%
  hc_add_series(data = tab01[, 2], name = "IV de 2018",
    dataLabels = list(enabled = TRUE), color = "#FA8072") %>%
  hc_add_series(data = tab01[, 3], name = "I de 2019",
    dataLabels = list(enabled = TRUE), color = "#E9967A") %>%
  hc_add_series(data = tab01[, 4], name = "Diferencia",
    dataLabels = list(enabled = TRUE), color = "teal") %>%
  hc_add_series(data = tab01[, 5], name = "Diferencia porcentual",
    dataLabels = list(enabled = TRUE, format = "{point.y} %"),
    color = "darkblue", yAxis = 1)

```

\begin{figure}[H]

Cifra de negocio de las 5 mayores tecnológicas



{

}

\caption{Gráfico del *dataframe*} \end{figure}

empresa	2019 (1º)	2018 (4º)	dif	difpct
Amazon	87400	72400	15000	20.7
Apple	91819	84310	7509	8.9
Facebook	21082	16914	4168	24.6
Google	46075	39276	6799	17.3
Microsoft	36906	32471	4435	13.7

empresa	2019 (1º)	2018 (4º)	dif	difpct
Amazon	87400	72400	15000	20.7
Apple	91819	84310	7509	8.9
Facebook	21082	16914	4168	24.6
Google	46075	39276	6799	17.3
Microsoft	36906	32471	4435	13.7

8.3 Estilo de las tablas

8.3.1 Estilo y posición

Vamos ahora a trabajar con la con la tabla y realizaremos algunos cambios sobre ella. Utilizaremos el paquete `kableExtra`. Este paquete permite trabajar con los *dataframe* para formatear la manera en que se van a mostrar en la pantalla. Existen multitud de paquetes que nos permitirían hacer cosas semejantes, entre los que podríamos destacar `formattable`, `DT` o `flextable` entre otros.

Nuestro primer cambio va a ser mostrar de forma diferente nuestra tabla. Con un estilo diferente a lo que ha aprecido hasta ahora. Le vamos a aplicar el estilo típico de **Bootstrap**, conocida librería JS con la que fue desarrollada Twitter. Te recomendamos visitar la viñeta del paquete en este sitio si quieres ver todas sus posibilidades.

```
kbl(tab01) %>%
  kable_styling()
```

Se puede apreciar que el aspecto es diferente al mostrado por las tablas hasta ahora. Este sería el formato más básico de *Bootstrap*. Existen otras opciones entre las que vamos a ir añadiendo algunas características, aplicables a los formatos.

Usando `paper...`

```
kbl(tab01) %>%
  kable_paper("hover")
```

Usando `classic...`

empresa	2019 (1º)	2018 (4º)	dif	difpct
Amazon	87400	72400	15000	20.7
Apple	91819	84310	7509	8.9
Facebook	21082	16914	4168	24.6
Google	46075	39276	6799	17.3
Microsoft	36906	32471	4435	13.7

empresa	2019 (1º)	2018 (4º)	dif	difpct
Amazon	87400	72400	15000	20.7
Apple	91819	84310	7509	8.9
Facebook	21082	16914	4168	24.6
Google	46075	39276	6799	17.3
Microsoft	36906	32471	4435	13.7

```
kbl(tab01) %>%
  kable_classic("hover")
```

Usando `minimal`...

```
kbl(tab01) %>%
  kable_minimal("hover")
```

Añadimos además del efecto *hover* con el ratón, el oscurecimiento o *striped* de las filas para facilitar la lectura y usabilidad.

```
kbl(tab01) %>%
  kable_material(c("striped", "hover"))
```

Finalizamos de nuevo con el primer formato que será el que mantendremos a partir de este momento.

```
kbl(tab01) %>%
  kable_styling(bootstrap_options = c("striped", "hover",
    "condensed", "responsive"))
```

empresa	2019 (1º)	2018 (4º)	dif	difpct
Amazon	87400	72400	15000	20.7
Apple	91819	84310	7509	8.9
Facebook	21082	16914	4168	24.6
Google	46075	39276	6799	17.3
Microsoft	36906	32471	4435	13.7

empresa	2019 (1°)	2018 (4°)	dif	difpct
Amazon	87400	72400	15000	20.7
Apple	91819	84310	7509	8.9
Facebook	21082	16914	4168	24.6
Google	46075	39276	6799	17.3
Microsoft	36906	32471	4435	13.7

Una de las características que en algunos casos puede ser muy valiosa, es la posibilidad de establecer la tabla como flotante a derecha o izquierda del texto.

```
kbl(tab01) %>%
  kable_styling(bootstrap_options = c("striped", "hover",
    "condensed", "responsive"), full_width = F, position = "float_right")
```

Este texto que estamos escribiendo, se ubicaría a la izquierda de la tabla, ya que hemos indicado que ésta debe ser situada a la derecha como elemento flotante. Al mismo tiempo se le ha

empresa	2019 (1°)	2018 (4°)	dif	difpct
Amazon	87400	72400	15000	20.7
Apple	91819	84310	7509	8.9
Facebook	21082	16914	4168	24.6
Google	46075	39276	6799	17.3
Microsoft	36906	32471	4435	13.7

incluido la posibilidad de que la tabla no ocupe el 100% del espacio sino que se autoajuste al tamaño de las columna. Este efecto puede ser muy utilizado para incluir comentarios sobre la tabla que estamos publicando. Como es lógico, la tabla puede ser flotante a la derecha o a la izquierda del texto.

Existen otras muchas características que pueden ser aplicadas para conseguir posicionar la tabla de la forma deseada:

- fijar la cabecera
- incluir la tabla en una caja
- aplicar tamaños de fuente
- ... y más.

8.3.2 Formato específico de fila o columna

Veamos algunas de las posibilidades que nos ofrece `kableExtra` para formatear partes de la tabla. Para ello debemos saber que las columnas se identifican por número secuencial (1 a n) al igual que las filas. Se pueden aplicar formatos tanto a filas como a columnas. Para ilustrar un ejemplo, vamos a utilizar una función de R muy frecuente denominada `ifelse()` que es el equivalente a la función `SI()` de Excel y con una estructura idéntica. Esta función admite la concatenación y sus parámetros son por este orden:

empresa	2019 (1º)	2018 (4º)	dif	difpct
Amazon	87400	72400	15000	20.7
Apple	91819	84310	7509	8.9
Facebook	21082	16914	4168	24.6
Google	46075	39276	6799	17.3
Microsoft	36906	32471	4435	13.7

empresa	2019 (1º)	2018 (4º)	dif	difpct
Amazon	87400	72400	15000	20.7
Apple	91819	84310	7509	8.9
Facebook	21082	16914	4168	24.6
Google	46075	39276	6799	17.3
Microsoft	36906	32471	4435	13.7

- condición que debe cumplirse
- resultado si verdadero
- resultado si falso

En nuestro caso nuestro resultado es la aplicación de un color verde o rojo según estemos por encima o por debajo de la media de la columna. Del mismo modo, también se puede aplicar como podemos observar un color directo a una columna determinada. Usaremos la función `column_spec()`.

```
kbl(tab01) %>%
  kable_styling(bootstrap_options = c("striped", "hover",
    "condensed", "responsive"), full_width = F) %>%
  column_spec(2, color = "teal", bold = TRUE) %>%
  column_spec(5, color = "white", background = ifelse(tab01$difpct >
    mean(tab01$difpct), "#3CB371", "#FA8072"), popover = paste(tab01[,
    1]))
```

Otra posibilidad es la de poder marcar celdas `cell_spec()` o filas con `row_spec()` en particular ...

```
kbl(tab01) %>%
  kable_styling(bootstrap_options = c("striped", "hover",
    "condensed", "responsive"), full_width = F) %>%
  row_spec(3:5, bold = T, color = "white", background = "teal")
```

Un caso particular para la fila 0, la de encabezado.

```
kbl(tab01) %>%
  kable_styling(c("striped", "hovered"), full_width = F) %>%
```