**Ejercicio 1**

En este ejercicio vamos a trabajar con datos de dos tablas relacionadas:

* Datos\_Basales (contiene información física como peso y altura).
* Pacientes (contiene información personal como nombres y fechas de nacimiento).

Queremos obtener los nombres y pesos de los pacientes cuyo peso es menor a 70 kg. Para resolver este ejercicio, vamos a utilizar la función left\_join() de la librería dplyr, que permite combinar tablas relacionadas por un campo común. El campo común en este caso es Paciente\_ID, que está presente en ambas tablas.

Pistas para resolver el ejercicio:

1. Filtra los datos en la tabla Datos\_Basales para obtener únicamente los pacientes con peso menor a 70 kg.
2. Combina estos datos con la tabla Pacientes usando left\_join() para obtener los nombres.
3. Selecciona solo las columnas relevantes: Nombre y Peso\_kg.

**Ejercicio 2**

En este ejercicio, trabajaremos con datos relacionados de las tablas:

* Visitas (contiene datos de las visitas médicas como presión arterial y glucosa).
* Pacientes (contiene información personal como nombres y fechas de nacimiento).

Queremos encontrar las visitas donde la presión sistólica sea mayor a 138. Para cada visita, mostraremos:

1. El nombre del paciente.
2. La fecha de la visita.
3. El valor de la presión sistólica.

Para combinar las tablas utilizaremos left\_join(), que nos permitirá conectar la tabla Visitas con la tabla Pacientes a través del campo común Paciente\_ID.

**Ejercicio 3**

En este ejercicio trabajaremos con datos relacionados de las tablas:

* Datos\_Basales (contiene información física como peso, altura e IMC).
* Pacientes (contiene información personal como nombres y fechas de nacimiento).

Queremos identificar a los pacientes que tienen un IMC superior a 30 (indicativo de obesidad). Para cada paciente, mostraremos:

1. Su nombre.
2. El valor de su IMC.

Además, utilizaremos left\_join() para conectar las tablas a través del campo común Paciente\_ID.

**Ejercicio 4**

En este ejercicio, trabajaremos con una API que proporciona datos diarios sobre COVID-19 en Estados Unidos. Usaremos el endpoint:

* <https://api.covidtracking.com/v1/us/daily.json>

1. Descargar los datos desde la API y convertirlos en un dataframe:

* Usaremos la función fromJSON() del paquete jsonlite para leer los datos desde la URL.
* Usa str() para explorar la estructura del dataframe y entender qué columnas están disponibles.

1. Realizar análisis básicos:

* Calcula el promedio de hospitalizaciones actuales (hospitalizedCurrently).
* Encuentra la fecha con el mayor número de muertes (death).
* Representa gráficamente la evolución de casos positivos (positive) a lo largo del tiempo con un gráfico de líneas.

Pistas para resolver el ejercicio:

* Paquetes necesario: Asegúrate de instalar y cargar el paquete jsonlite con install.packages("jsonlite") y library(jsonlite).
* Cargar los datos: Usa la función fromJSON(url) para leer los datos directamente desde la API.
* Estructurar datos: Convierte las fechas en un formato adecuado para gráficos o análisis con: as.Date(as.character(covid\_data$date), format = "%Y%m%d")
* Explorar los datos: Usa str(nombre\_del\_dataframe) para ver la estructura de los datos y familiarizarte con las columnas disponibles.
* Evita error con valores faltantes: Para ignorar valores vacíos, en los cálculos como mean() o max(), usa el argumento na.rm = TRUE. Ejemplo: mean(covid\_data$deathIncrease, na.rm = TRUE)
* Análisis y gráficos: Usa funciones básicas de R para cálculos (mean() y max()) y graficado con el paquete ggplot2.