Laboratorio 1: Introducción al tidyverse

Ejercicio 1

Para este ejercicio vamos a trabajar con los datos `ChickWeight`, para tenerlos disponibles simplemente hacemos `data(ChickWeight)`

Los datos `ChickWeight` tienen 578 filas y 4 columnas, provienen de un experimento sobre el efecto de la dieta en el crecimiento temprano de los pollos. Se puede obtener más información sobre los datos mediante `?ChickWeight`

Datos completos

Cada pollo debe tener 12 observaciones, usá el paquete `dplyr` para identificar cuántos pollos tienen efectivamente los datos completos.

1. ¿Cuántos pollos tienen menos de 12 observaciones?

```
dim(pollo %>%
      group_by(Chick) %>%
      summarise(obs=n()) %>%
      filter(obs<12))[1]
```

```
## [1] 5
```

2. Extrae un subconjunto de los datos correspondientes a los pollos con información completa (12 observaciones) y guardalos en un objeto llamado `complete`.

```
complete <- left_join(pollo, (pollo %>% group_by(Chick) %>% summarise(obs=n())), by="Chick") %>% filter
unique(complete$obs)
```

```
## [1] 12
```

(ayuda: usa `mutate` para crear una variable auxiliar con el número de observaciones)

Crear variable peso ganado

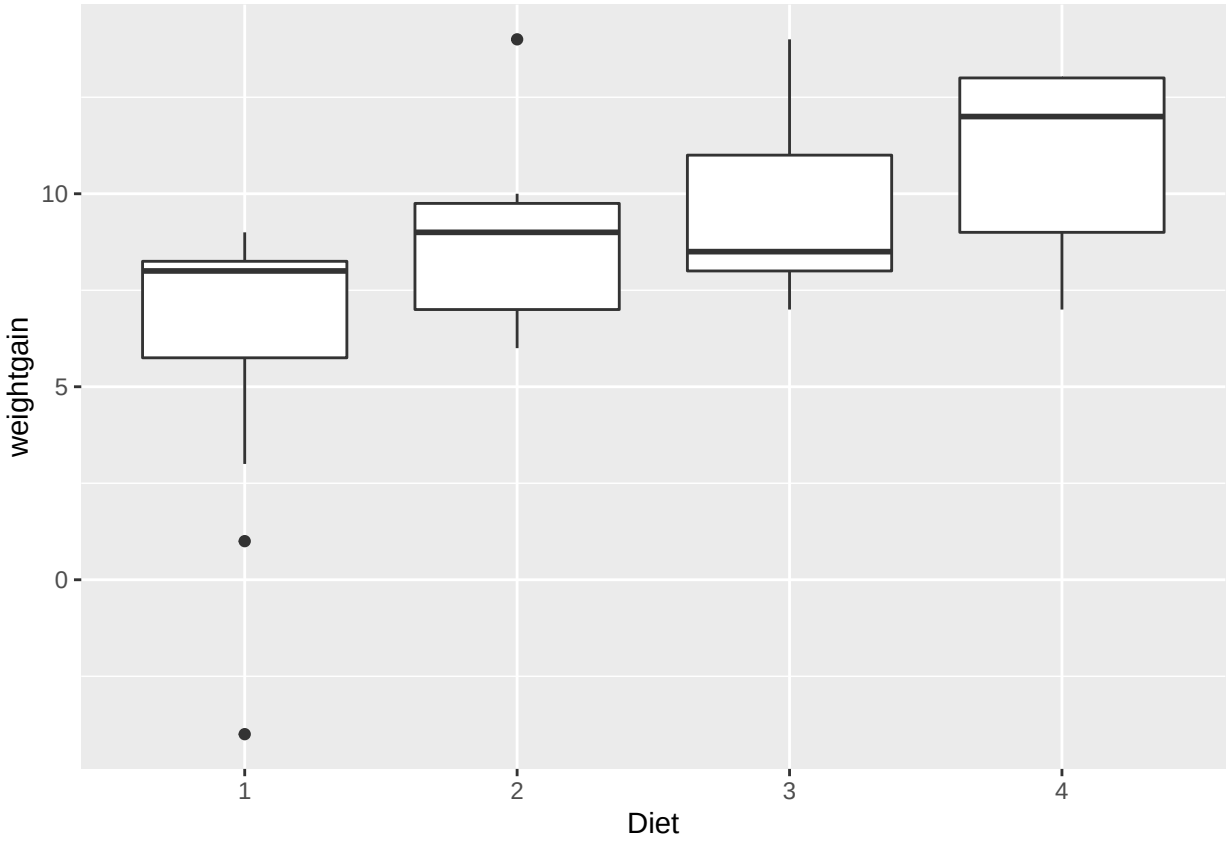
Con los datos completos, crea una nueva variable que mide la diferencia del peso en cada momento respecto del momento inicial. Llama esta variable `weightgain`.

```
complete <- left_join(complete,
                      (complete %>% filter(Time==0) %>%
                         rename(Inicial = weight) %>%
                         select(Inicial, Chick)),
                      by="Chick") %>%
      mutate(weightgain = weight-Inicial)
```

Dibujar boxplot

1. Usando `ggplot2` crear un gráfico de boxplots de la variable `weightgain` contra `Diet` para el día 2.

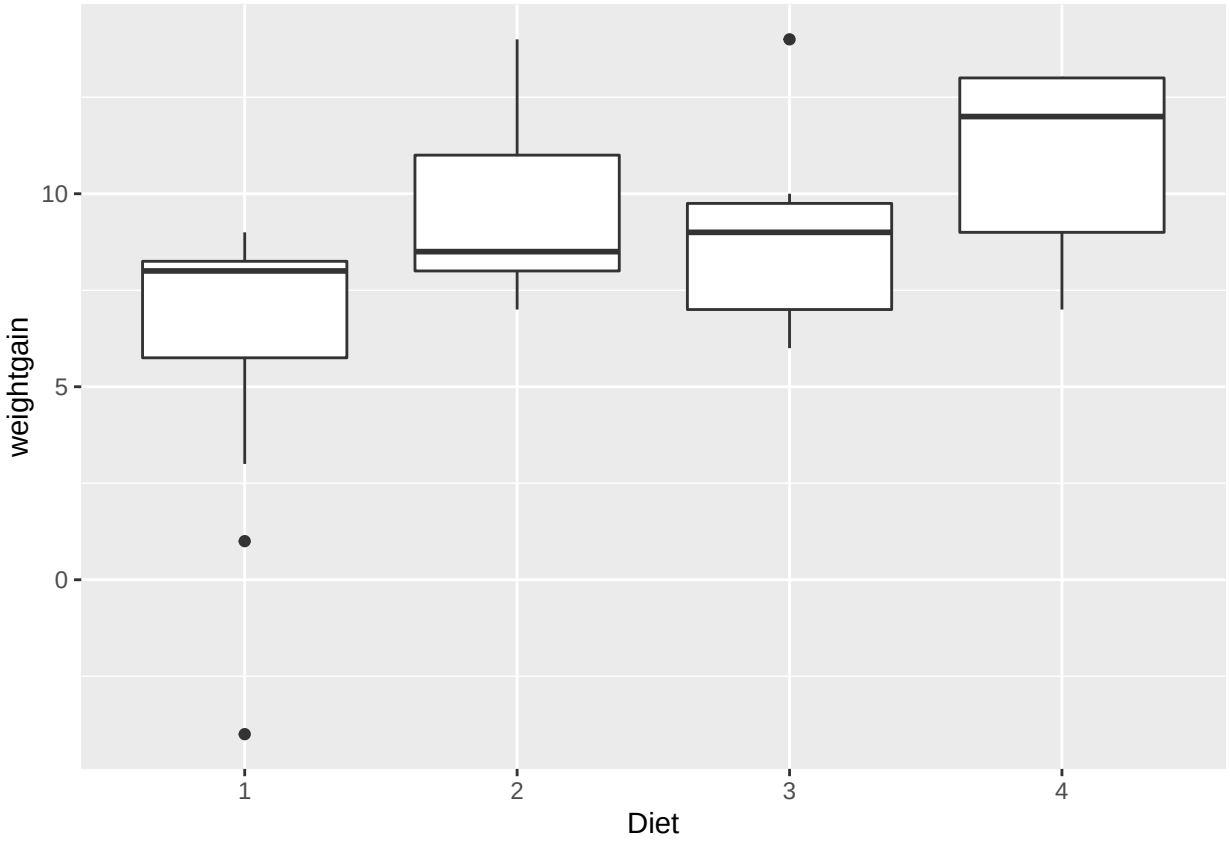
```
ggplot(complete %>% filter(Time == 2)) +
  geom_boxplot(aes(x=Diet, y=weightgain))
```



2. Describe la relación en un par de frases.

En el gráfico cambia el orden de las categorías `Diet` de para que los boxplots queden ordenados según la mediana de `weightgain`.

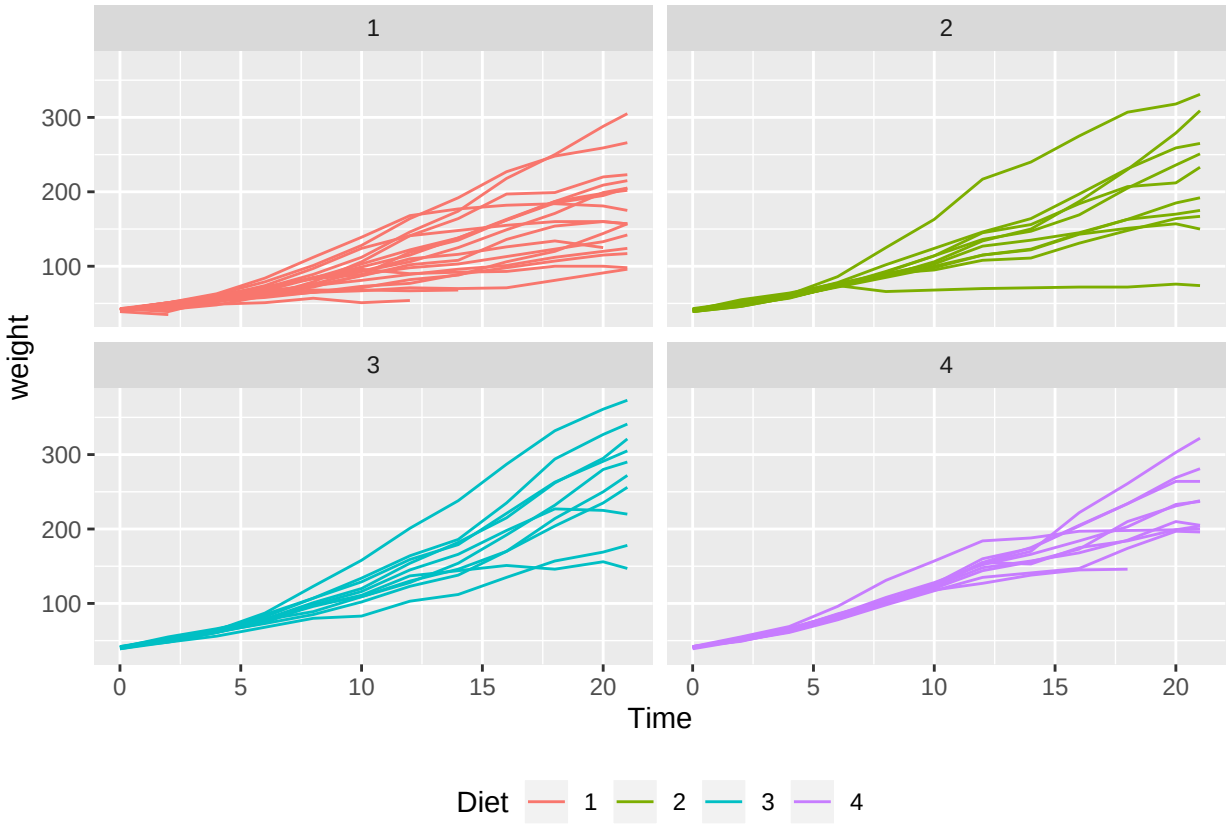
```
ggplot(complete %>% filter(Time == 2)) +
      geom_boxplot(aes(x=Diet, y=weightgain)) +
      scale_x_discrete(limits = as.numeric(((
        complete %>%
          filter(Time == 2) %>%
          group_by(Diet) %>%
          summarize(mediana=median(weightgain))) %>%
          arrange(mediana))$Diet))
```



Graficar evolución del peso

Crear un dibujo que tenga: 1. `Time` en el eje x y `weight` en el eje y. 2. Una linea para cada pollo
3. Divide en facets y colorea seg?n `Diet`.
4. Incluye la leyenda debajo del dibujo (con `theme`)

```
ggplot(pollo) +
      geom_line(aes(x=Time, y=weight, group=Chick, colour=Diet)) +
      facet_wrap(~Diet) +
      theme(legend.position="bottom")
```



Graficar solo pollos más grandes

1. Seleccionar el pollo con máximo peso en el momento 21 para cada dieta.

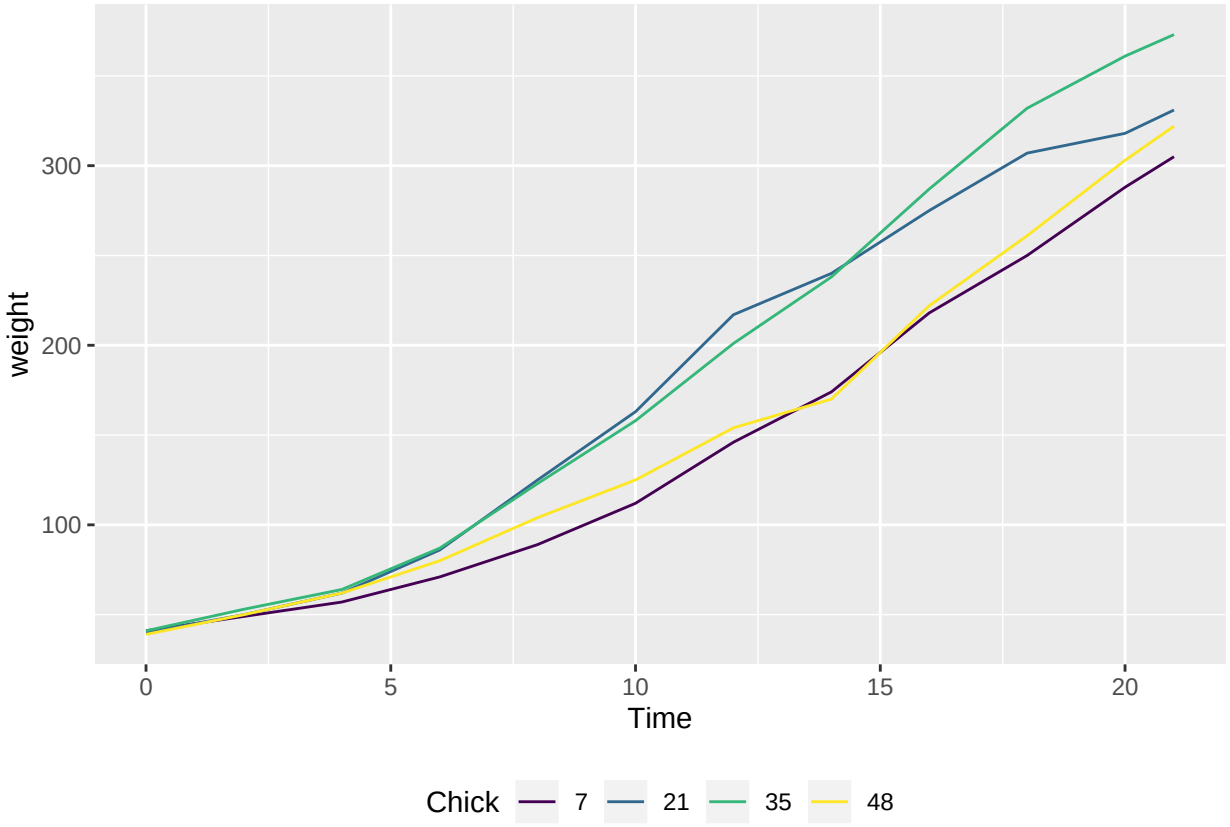
```
maximos <- pollo %>%
  filter(Time == 21) %>%
  group_by(Diet) %>%
  summarize(Mayor_peso=max(weight)) # Consigo el máximo peso por dieta para Time == 21
pollo <- left_join(pollo, maximos, by="Diet") %>%
  mutate(dif=weight-Mayor_peso)
# Los agrego a la base "pollo" y calculo la diferencia con weight. Aquellos que tengan diferencia 0, se
table(pollo$dif == 0) # Verifico que sea uno solo pollo por dieta (es decir, que haya solo 4 ceros)
```

```
##
## FALSE  TRUE
##   574     4
```

```
pollitos.gorditos <- pollo[which(pollo$dif == 0), "Chick"] # Identifico los pollos
```

2. Repetir el dibujo anterior solo con estos 4 pollos y sin facetas.

```
ggplot(pollo %>% filter(Chick %in% pollitos.gorditos)) +
      geom_line(aes(x=Time, y=weight, color=Chick)) +
      theme(legend.position="bottom")
```

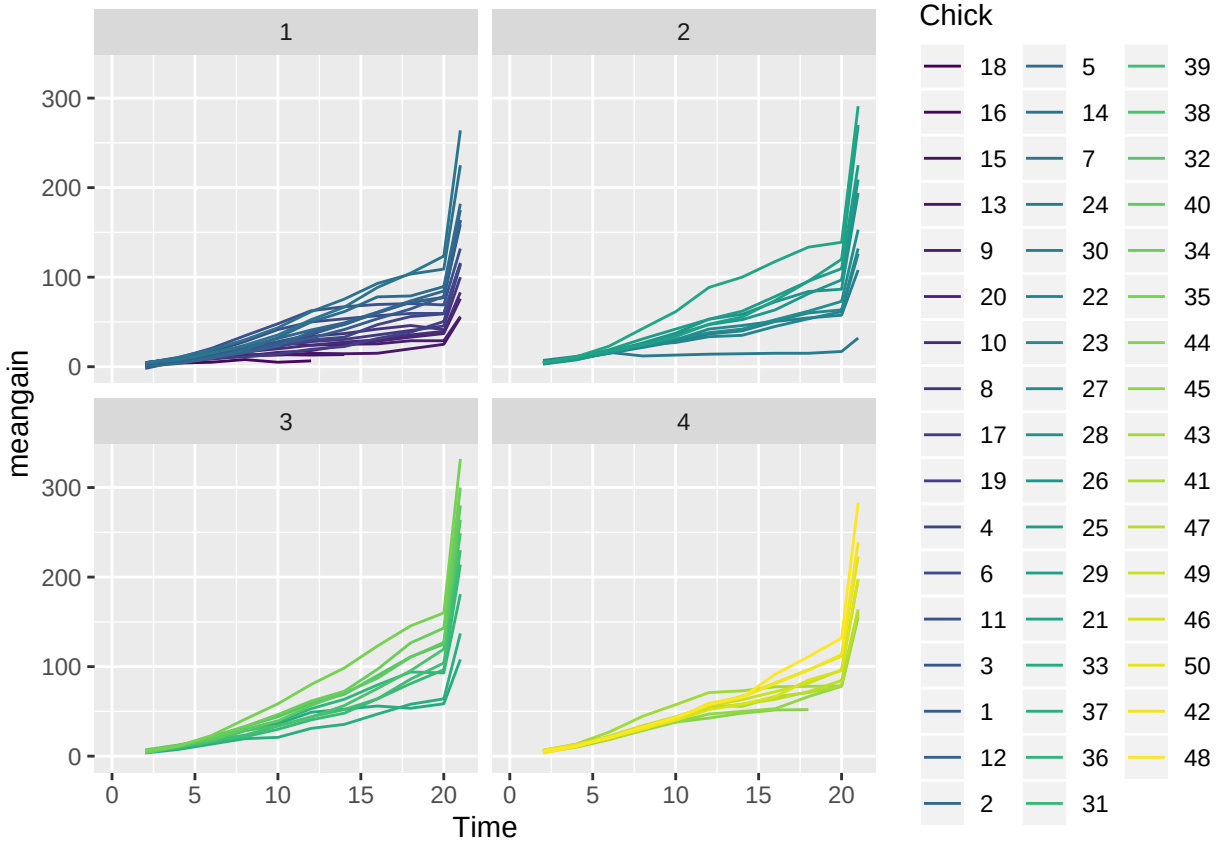


Ganancia de peso promedio

Calcular la ganancia de peso media diaria y repetir el dibujo.

```
pollo <- select(pollo, -Mayor_peso, -dif)
pollo <- pollo %>% group_by(Chick) %>% mutate(dif=Time-lag(Time, default=0))
pollo <- left_join(pollo,
                   (pollo %>%
                      filter(Time==0) %>%
                      rename(Inicial=weight) %>%
                      select(Inicial, Chick)),
                   by="Chick") %>%
  mutate(weightgain = weight-Inicial)
pollo <- pollo %>% mutate(meangain=weightgain/dif)
ggplot(pollo) +
      geom_line(aes(x=Time, y=meangain, color=Chick)) +
      facet_wrap(~Diet)
```

```
## Warning: Removed 50 rows containing missing values (geom_path).
```



Ejercicio 2

Crear una función de R que:

1. Ajuste un modelo linear

```
modelolineal <- function(x, y, data) {lm(y ~ x, data=data)}
```

2. Dibuje los residuos contra valores ajustados del modelo

Los argumentos de la función deben ser:
`x` (variable explicativa),
`y` (variable de respuesta)
`col` color para la linea horizontal

La etiqueta del eje x debe ser 'Valores ajustados' y la etiqueta del eje y debe ser 'Residuos'. Prueba como funciona usando `data(LifeCycleSavings)`, con `x = sr`, `y = ddpi` y `col = red`.