Tidyverse - Exercices (Correction)

October 2020

Iris

Nous considérons le jeu de données `iris`. Répondre aux questions suivantes en utilisant les fonctions du package `dplyr` :

1. Sélectionner les variables `Petal.Width` et `Species`.

```
iris %>%
  select(Petal.Width, Species) %>%
  head(3)
```

```
##   Petal.Width Species
## 1         0.2  setosa
## 2         0.2  setosa
## 3         0.2  setosa
```

2. Construire une table qui contient uniquement les iris d'espèce `versicolor` ou `virginica`.

```
# Avec l'opérateur %in%
iris %>%
  filter(Species %in% c("versicolor", "virginica")) %>%
  head(3)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 1          7.0         3.2          4.7         1.4 versicolor
## 2          6.4         3.2          4.5         1.5 versicolor
## 3          6.9         3.1          4.9         1.5 versicolor
```

```
# Avec l'opérateur logique OR
iris %>%
  filter(Species == "versicolor" | Species == "virginica") %>%
  head(3)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 1          7.0         3.2          4.7         1.4 versicolor
## 2          6.4         3.2          4.5         1.5 versicolor
## 3          6.9         3.1          4.9         1.5 versicolor
```

3. Calculer the nombre d'iris `setosa` en utilisant `summarise`.

```
iris %>%
  filter(Species == "setosa") %>%
  summarise(n = n())
```

```
##    n
## 1 50
```

4. Calculer la moyenne de la variable `Petal.Width` pour les iris de l'espèce `versicolor`.

```r
iris %>%
  filter(Species == "versicolor") %>%
  summarise(mean_petal_width = mean(Petal.Width))
```

```
##   mean_petal_width
## 1            1.326
```

5. Ajouter dans le jeu de données la variable `Sum_Petal` qui correspond à la somme de `Petal.Width` et `Sepal.Width`.

```r
iris %>%
  mutate(Sum_Petal = Petal.Width + Sepal.Width) %>%
  head(3)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species Sum_Petal
## 1          5.1         3.5          1.4         0.2  setosa       3.7
## 2          4.9         3.0          1.4         0.2  setosa       3.2
## 3          4.7         3.2          1.3         0.2  setosa       3.4
```

6. Calculer la moyenne et la variance de la variable `Sepal.Length` pour chaque espèce.

```r
iris %>%
  group_by(Species) %>%
  summarise(mean_sepal_length = mean(Sepal.Length),
            var_sepal_length = var(Sepal.Length),
            .groups = 'drop') # Évite un message d'avertissement
```

```
## # A tibble: 3 x 3
##   Species    mean_sepal_length var_sepal_length
##   <fct>                  <dbl>            <dbl>
## 1 setosa                  5.01            0.124
## 2 versicolor              5.94            0.266
## 3 virginica               6.59            0.404
```

Aviation

Nous considérons la table `hflights` qui contient des informations sur les vols au départ des aéroports *Houston George Bush Intercontinental Airport* (IATA: IAH) et *William P. Hobby Airport* (IATA: HOU),

```r
library(hflights)
hflights <- as_tibble(hflights)
```

1. Sélectionner les variables qui se situent entre `Origin` et `Cancelled` de différentes façons.

```r
# Remarque : positions de Origin et Cancelled
names(hflights)[c(14, 19)]
```

```
## [1] "Origin"    "Cancelled"
```

```r
# Remarque : quelles sont les variables ?
names(hflights)[13:18]
```

```
## [1] "DepDelay" "Origin"   "Dest"     "Distance" "TaxiIn"   "TaxiOut"
```

```r
# Par le nom
hflights %>%
  select(Dest, Distance, TaxiIn, TaxiOut) %>%
  head(3)
```

```
## # A tibble: 3 x 4
```

```
##   Dest  Distance TaxiIn TaxiOut
##   <chr>    <int>  <int>   <int>
## 1 DFW        224      7      13
## 2 DFW        224      6       9
## 3 DFW        224      5      17
```

```
# Par la position (argument vectoriel)
hflights %>%
  select(names(hflights)[15:18]) %>%
  head(3)
```

```
## # A tibble: 3 x 4
##   Dest  Distance TaxiIn TaxiOut
##   <chr>    <int>  <int>   <int>
## 1 DFW        224      7      13
## 2 DFW        224      6       9
## 3 DFW        224      5      17
```

```
# Par des helpers
hflights %>%
  select(matches("D?st.*") | starts_with("Taxi")) %>%
  head(3)
```

```
## # A tibble: 3 x 4
##   Dest  Distance TaxiIn TaxiOut
##   <chr>    <int>  <int>   <int>
## 1 DFW        224      7      13
## 2 DFW        224      6       9
## 3 DFW        224      5      17
```

2. Sélectionner les variables `DepTime`, `ArrTime`, `ActualElapsedTime`, `AirTime`, `ArrDelay` et `DepDelay`.

```
hflights %>%
  select(ends_with(c("Time", "Delay"))) %>%
  head(3)
```

```
## # A tibble: 3 x 6
##   DepTime ArrTime ActualElapsedTime AirTime ArrDelay DepDelay
##     <int>   <int>             <int>   <int>    <int>    <int>
## 1    1400    1500                60      40      -10        0
## 2    1401    1501                60      45       -9        1
## 3    1352    1502                70      48       -8       -8
```

3. Ajouter une variable `ActualGroundTime` qui correspond à `ActualElapsedTime` moins `AirTime`.

```
hflights %>%
  mutate(ActualGroundTime = ActualElapsedTime - AirTime) %>%
  select(ActualElapsedTime, AirTime, ActualGroundTime) %>%
  head(3)
```

```
## # A tibble: 3 x 3
##   ActualElapsedTime AirTime ActualGroundTime
##               <int>   <int>            <int>
## 1                60      40               20
## 2                60      45               15
## 3                70      48               22
```

4. Ajouter une variable `AverageSpeed` qui donne la vitesse moyenne du vol et ordonner la table selon les valeurs décroissantes de cette variable.

```
hflights %>%
  mutate(AverageSpeed = Distance / AirTime) %>%
  select(Origin, Dest, Distance, AirTime, AverageSpeed) %>%
  arrange(desc(AverageSpeed)) %>%
  head(3)
```

```
## # A tibble: 3 x 5
##   Origin Dest  Distance AirTime AverageSpeed
##   <chr>  <chr>    <int>   <int>        <dbl>
## 1 IAH    AUS        140      11         12.7
## 2 IAH    MEM        469      42         11.2
## 3 IAH    CLT        913      85         10.7
```

5. Sélectionner les vols à destination de `JFK`.

```
hflights %>%
  filter(Dest == "JFK") %>%
  select(FlightNum, Origin, Dest) %>%
  head(3)
```

```
## # A tibble: 3 x 3
##   FlightNum Origin Dest
##       <int> <chr>  <chr>
## 1       620 HOU    JFK
## 2       622 HOU    JFK
## 3       620 HOU    JFK
```

6. Calculer le nombre de vols à destination de `JFK`.

```
hflights %>%
  filter(Dest == "JFK") %>%
  summarise(n = n())
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   695
```

7. Créer un résumé de `hflights` qui contient :
   - `n` : le nombre total de vols ;
   - `n_dest`: le nombre total de destinations ;
   - `n_carrier` : le nombre total de compagnies.

```
hflights %>%
  summarise(n         = n(),
            n_dest    = n_distinct(Dest),
            n_carrier = n_distinct(UniqueCarrier))
```

```
## # A tibble: 1 x 3
##        n n_dest n_carrier
##    <int>  <int>     <int>
## 1 227496    116        15
```

8. Créer un résumé de `hflights` qui contient, pour les vols de la compagnie **AA** :
   - le nombre total de vols ;
   - le nombre total de vols annulés ;
   - la valeur moyenne de `ArrDelay` (attention à la gestion des `NA`).

```r
hflights %>%
  filter(UniqueCarrier == "AA") %>%
  summarise(n           = n(),
            n_cancelled = sum(Cancelled),
            mean_delay  = mean(ArrDelay, na.rm = TRUE))
```

```
## # A tibble: 1 x 3
##       n n_cancelled mean_delay
##   <int>       <int>      <dbl>
## 1  3244          60      0.892
```

9. Calculer pour chaque compagnie :
   - le nombre total de vols ;
   - La valeur moyenne de `AirTime`.

```r
hflights %>%
  group_by(UniqueCarrier) %>%
  summarise(n             = n(),
            mean_air_time = mean(AirTime, na.rm = TRUE),
            .groups = 'drop') # Évite un message d'avertissement
```

```
## # A tibble: 15 x 3
##    UniqueCarrier     n mean_air_time
##    <chr>         <int>         <dbl>
##  1 AA             3244          69.7
##  2 AS              365         254.
##  3 B6              695         184.
##  4 CO            70032         145.
##  5 DL             2641          97.8
##  6 EV             2204         104.
##  7 F9              838         125.
##  8 FL             2139          92.7
##  9 MQ             4648          93.8
## 10 OO            16061         113.
## 11 UA             2072         157.
## 12 US             4082         134.
## 13 WN            45343          86.7
## 14 XE            73053          83.2
## 15 YV               79         122.
```

10. Ordonner les compagnies en fonction des retards moyens au départ.

```r
hflights %>%
  group_by(UniqueCarrier) %>%
  summarise(mean_dep_delay = mean(DepDelay, na.rm = TRUE),
            .groups = 'drop') %>% # Évite un message d'avertissement
  arrange(mean_dep_delay)
```

```
## # A tibble: 15 x 2
##    UniqueCarrier mean_dep_delay
##    <chr>                  <dbl>
##  1 YV                      1.54
##  2 US                      1.62
##  3 AS                      3.71
##  4 FL                      4.72
##  5 F9                      5.09
```

```
##  6 AA                     6.39
##  7 XE                     7.71
##  8 OO                     8.89
##  9 CO                     9.26
## 10 DL                     9.37
## 11 MQ                    11.1
## 12 EV                    12.5
## 13 UA                    12.9
## 14 B6                    13.3
## 15 WN                    13.5
```

Tennis

Nous considérons les données sur les résultats de tennis dans les tournois du Grand Chelem en 2013. Les données, ainsi que le descriptif des variables, se trouvent à l'adresse suivante :

https://archive.ics.uci.edu/ml/datasets/Tennis+Major+Tournament+Match+Statistics

Nous considérons d'abord le tounoi masculin de Roland Garros. Utiliser les verbes de `dplyr` pour répondre aux questions suivantes.

1. Importer les données.

```
fpath <- file.path("data", "Tennis", "FrenchOpen-men-2013.csv")
rg_tbl <- read_csv(fpath)
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   Player1 = col_character(),
##   Player2 = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
rg_tbl %>% glimpse()
```

```
## Rows: 125
## Columns: 42
## $ Player1 <chr> "Pablo Carreno-Busta", "Somdev Devvarman", "Tobias Kamke", ...
## $ Player2 <chr> "Roger Federer", "Daniel Munoz-De La Nava", "Paolo Lorenzi"...
## $ Round   <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ Result  <dbl> 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0,...
## $ FNL.1   <dbl> 0, 3, 3, 3, 0, 3, 2, 3, 0, 0, 3, 2, 0, 3, 3, 3, 0, 2, 3, 2,...
## $ FNL.2   <dbl> 3, 0, 2, 1, 3, 1, 3, 2, 3, 3, 0, 3, 3, 0, 2, 0, 3, 3, 0, 2,...
## $ FSP.1   <dbl> 62, 62, 62, 72, 52, 70, 63, 59, 56, 63, 48, 78, 66, 83, 69,...
## $ FSW.1   <dbl> 27, 54, 53, 87, 31, 58, 71, 42, 27, 62, 29, 85, 48, 51, 84,...
## $ SSP.1   <dbl> 38, 38, 38, 28, 48, 30, 37, 41, 44, 37, 52, 22, 34, 17, 31,...
## $ SSW.1   <dbl> 11, 22, 15, 19, 22, 18, 38, 25, 13, 29, 20, 22, 13, 6, 29, ...
## $ ACE.1   <dbl> 1, 7, 4, 14, 4, 4, 5, 7, 0, 5, 12, 2, 7, 6, 11, 5, 1, 5, 7,...
## $ DBF.1   <dbl> 3, 3, 6, 2, 4, 4, 5, 2, 6, 4, 4, 1, 5, 0, 4, 0, 5, 7, 2, 3,...
## $ WNR.1   <dbl> 12, 26, 42, 48, 21, 35, 45, 41, 12, 41, 33, 24, 24, 27, 57,...
## $ UFE.1   <dbl> 29, 20, 55, 27, 24, 36, 80, 49, 28, 44, 12, 30, 24, 18, 42,...
## $ BPC.1   <dbl> 1, 5, 10, 4, 1, 6, 5, 10, 1, 1, 7, 5, 1, 5, 6, 6, 5, 4, 7, ...
## $ BPW.1   <dbl> 3, 8, 22, 13, 1, 12, 12, 14, 2, 6, 11, 17, 5, 14, 7, 10, 8,...
## $ NPA.1   <dbl> 9, 12, 14, 14, 3, 8, 28, 11, 11, 19, 14, 12, 11, 5, 8, 13, ...
## $ NPW.1   <dbl> 20, 21, 32, 30, 5, 10, 41, 18, 18, 27, 21, 18, 14, 6, 13, 1...
## $ TPW.1   <dbl> 50, 120, 140, 163, 72, 130, 160, 136, 54, 132, 91, 174, 85,...
```

```
## $ ST1.1   <dbl> 2, 6, 6, 7, 3, 6, 3, 3, 1, 6, 6, 6, 4, 6, 6, 6, 4, 4, 6, 5,...
## $ ST2.1   <dbl> 2, 6, 6, 6, 4, 5, 6, 1, 2, 6, 6, 6, 2, 6, 4, 6, 3, 6, 6, 6,...
## $ ST3.1   <dbl> 3, 7, 3, 5, 4, 6, 3, 6, 3, 6, 6, 6, 5, 6, 7, 6, 4, 3, 6, 4,...
## $ ST4.1   <dbl> NA, NA, 0, 7, NA, 6, 7, 6, NA, NA, NA, 4, NA, NA, 4, NA, NA...
## $ ST5.1   <dbl> NA, NA, 6, NA, NA, NA, 5, 7, NA, NA, NA, 4, NA, NA, 6, NA, ...
## $ FSP.2   <dbl> 68, 52, 46, 53, 58, 68, 59, 49, 50, 56, 47, 52, 66, 55, 54,...
## $ FSW.2   <dbl> 33, 35, 42, 58, 39, 48, 67, 40, 26, 64, 22, 67, 44, 36, 58,...
## $ SSP.2   <dbl> 32, 48, 54, 47, 42, 32, 41, 51, 50, 44, 53, 48, 34, 45, 46,...
## $ SSW.2   <dbl> 14, 24, 31, 38, 19, 17, 27, 29, 20, 35, 15, 47, 17, 18, 36,...
## $ ACE.2   <dbl> 10, 0, 6, 13, 10, 5, 5, 4, 5, 10, 3, 12, 10, 8, 7, 4, 3, 5,...
## $ DBF.2   <dbl> 0, 2, 8, 10, 1, 5, 6, 6, 1, 7, 2, 5, 6, 5, 8, 4, 4, 4, 4, 3...
## $ WNR.2   <dbl> 33, 40, 39, 72, 42, 30, 54, 44, 35, 46, 23, 72, 46, 32, 60,...
## $ UFE.2   <dbl> 19, 47, 54, 56, 37, 51, 57, 72, 15, 38, 31, 64, 33, 29, 71,...
## $ BPC.2   <dbl> 7, 1, 10, 4, 4, 1, 6, 9, 7, 1, 2, 5, 5, 1, 5, 1, 8, 6, 1, 4...
## $ BPW.2   <dbl> 7, 16, 18, 13, 7, 7, 20, 10, 12, 3, 2, 10, 11, 3, 15, 4, 11...
## $ NPA.2   <dbl> 14, 22, 19, 33, 12, 6, 14, 19, 13, 9, 9, 11, 10, 12, 24, 9,...
## $ NPW.2   <dbl> 18, 25, 27, 43, 13, 9, 22, 35, 21, 20, 22, 18, 16, 20, 33, ...
## $ TPW.2   <dbl> 88, 106, 139, 149, 93, 93, 175, 120, 92, 130, 59, 177, 108,...
## $ ST1.2   <dbl> 6, 3, 3, 6, 6, 2, 6, 6, 6, 7, 4, 4, 6, 3, 4, 2, 6, 6, 1, 7,...
## $ ST2.2   <dbl> 6, 3, 3, 3, 6, 7, 2, 6, 6, 7, 2, 4, 6, 4, 6, 2, 6, 3, 3, 2,...
## $ ST3.2   <dbl> 6, 5, 6, 7, 6, 0, 6, 4, 6, 7, 2, 7, 7, 2, 6, 3, 6, 6, 2, 6,...
## $ ST4.2   <dbl> NA, NA, 6, 6, NA, 4, 5, 1, NA, NA, NA, 6, NA, NA, 6, NA, NA...
## $ ST5.2   <dbl> NA, NA, 3, NA, NA, NA, 7, 5, NA, NA, NA, 6, NA, NA, 2, NA, ...
```

2. Afficher le nom des adversaires de Roger Federer.

```
rg_tbl %>%
  filter(Player2 == "Roger Federer") %>% # Roger Federer n'est jamais Player1
  select(Player1)
```

```
## # A tibble: 5 x 1
##   Player1
##   <chr>
## 1 Pablo Carreno-Busta
## 2 Somdev Devvarman
## 3 Julien Benneteau
## 4 Gilles Simon
## 5 Jo-Wilfried Tsonga
```

3. Afficher le nom des demi-finalistes.

```
rg_tbl %>%
  filter(Round == 6) %>% # 7: Finale, 6, Demi-finale, ...
  select(Player1, Player2)
```

```
## # A tibble: 2 x 2
##   Player1        Player2
##   <chr>          <chr>
## 1 David Ferrer   Jo-Wilfried Tsonga
## 2 Novak Djokovic Rafael Nadal
```

4. Combien y a t-il eu de points disputés en moyenne par match ? Il faudra penser à ajouter dans la table une variable correspondant au nombre de points de chaque match (verbe `mutate`).

```
rg_tbl %>%
  mutate(total_points = TPW.1 + TPW.2) %>%
  summarise(mean_total_points = mean(total_points))
```

```
## # A tibble: 1 x 1
##   mean_total_points
##               <dbl>
## 1              219.
```

5. Combien y a t-il eu d'aces par match en moyenne ?

```
rg_tbl %>%
  mutate(aces = ACE.1 + ACE.2) %>%
  summarise(mean_aces = mean(aces))
```

```
## # A tibble: 1 x 1
##   mean_aces
##       <dbl>
## 1      12.7
```

6. Combien y a t-il eu d'aces par match en moyenne à chaque tour ?

```
rg_tbl %>%
  mutate(aces = ACE.1 + ACE.2) %>%
  group_by(Round) %>%
  summarise(mean_aces = mean(aces))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 7 x 2
##   Round mean_aces
##   <dbl>     <dbl>
## 1     1     13.5
## 2     2     13.2
## 3     3     12.6
## 4     4      9.12
## 5     5      7
## 6     6     10
## 7     7      6
```

7. Combien y a t-il eu de doubles fautes au total dans le tournoi ?

```
rg_tbl %>%
  mutate(double_faults = DBF.1 + DBF.2) %>%
  summarise(sum_double_faults = sum(double_faults, na.rm = TRUE))
```

```
## # A tibble: 1 x 1
##   sum_double_faults
##               <dbl>
## 1               812
```

8. Importer les données pour le tournoi de Wimbledon masculin de 2013.

```
fpath <- file.path("data", "Tennis", "Wimbledon-men-2013.csv")
w_tbl <- read_csv(fpath)
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   Player1 = col_character(),
##   Player2 = col_character(),
##   TPW.1 = col_logical(),
##   TPW.2 = col_logical()
```

```
## )
## See spec(...) for full column specifications.
```

```
w_tbl %>% glimpse()
```

```
## Rows: 114
## Columns: 42
## $ Player1 <chr> "B.Becker", "J.Ward", "N.Mahut", "T.Robredo", "R.Haase", "M...
## $ Player2 <chr> "A.Murray", "Y-H.Lu", "J.Hajek", "A.Bogomolov Jr.", "M.Youz...
## $ Round   <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ Result  <dbl> 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0,...
## $ FNL.1   <dbl> 0, 1, 3, 3, 0, 0, 3, 0, 0, 3, 1, 3, 1, 3, 0, 3, 0, 0, 3, 1,...
## $ FNL.2   <dbl> 3, 3, 0, 0, 3, 3, 1, 3, 3, 0, 3, 0, 3, 1, 3, 0, 3, 3, 2, 3,...
## $ FSP.1   <dbl> 59, 62, 72, 77, 68, 59, 63, 61, 61, 67, 64, 78, 69, 63, 66,...
## $ FSW.1   <dbl> 29, 77, 44, 40, 61, 41, 56, 47, 31, 56, 66, 46, 60, 58, 48,...
## $ SSP.1   <dbl> 41, 38, 28, 23, 32, 41, 37, 39, 39, 33, 36, 22, 31, 37, 34,...
## $ SSW.1   <dbl> 14, 35, 10, 12, 15, 27, 21, 21, 16, 21, 18, 9, 14, 21, 21, ...
## $ ACE.1   <dbl> 5, 18, 17, 6, 7, 7, 21, 3, 4, 22, 13, 6, 3, 20, 5, 18, 3, 1...
## $ DBF.1   <dbl> 1, 4, 3, 0, 2, 6, 3, 1, 5, 6, 2, 2, 2, 7, 5, 2, 1, 3, 10, 2...
## $ WNR.1   <dbl> 26, 60, 41, 25, 32, 22, 56, 28, 20, 61, 55, 19, 33, 64, 42,...
## $ UFE.1   <dbl> 18, 28, 18, 11, 29, 28, 32, 16, 18, 29, 40, 20, 33, 29, 30,...
## $ BPC.1   <dbl> 5, 13, 8, 14, 2, 6, 16, 4, 1, 8, 3, 14, 7, 11, 11, 6, 0, 2,...
## $ BPW.1   <dbl> 1, 1, 5, 5, 0, 1, 4, 0, 1, 3, 1, 6, 2, 6, 2, 3, 0, 0, 4, 4,...
## $ NPA.1   <dbl> 28, 27, 26, 14, 29, 11, 21, 33, 14, 47, 22, 9, 34, 25, 49, ...
## $ NPW.1   <dbl> 19, 19, 17, 11, 20, 6, 15, 24, 9, 35, 15, 7, 25, 18, 29, 32...
## $ TPW.1   <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
## $ ST1.1   <dbl> 4, 7, 6, 6, 4, 3, 6, 3, 3, 7, 5, 6, 4, 6, 6, 7, 3, 4, 6, 7,...
## $ ST2.1   <dbl> 3, 4, 6, 6, 5, 2, 6, 4, 4, 6, 4, 6, 7, 6, 4, 6, 2, 0, 4, 5,...
## $ ST3.1   <dbl> 2, 6, 6, 6, 5, 6, 3, 6, 4, 6, 7, 6, 4, 6, 5, 6, 0, 4, 7, 3,...
## $ ST4.1   <dbl> NA, 6, NA, NA, NA, NA, 6, NA, NA, NA, 2, NA, 2, 6, NA, NA, ...
## $ ST5.1   <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
## $ FSP.2   <dbl> 57, 67, 70, 79, 67, 70, 73, 71, 70, 54, 67, 58, 63, 63, 65,...
## $ FSW.2   <dbl> 39, 85, 34, 35, 53, 56, 59, 55, 45, 40, 64, 33, 65, 48, 59,...
## $ SSP.2   <dbl> 43, 33, 30, 21, 33, 30, 27, 29, 30, 46, 33, 42, 37, 37, 35,...
## $ SSW.2   <dbl> 20, 31, 14, 8, 17, 11, 14, 16, 16, 22, 23, 17, 26, 24, 20, ...
## $ ACE.2   <dbl> 11, 12, 4, 1, 9, 25, 7, 15, 16, 4, 16, 4, 12, 7, 20, 2, 7, ...
## $ DBF.2   <dbl> 2, 3, 0, 4, 3, 3, 8, 2, 2, 2, 0, 12, 6, 1, 6, 3, 0, 2, 13, ...
## $ WNR.2   <dbl> 38, 57, 24, 16, 40, 53, 33, 40, 41, 22, 52, 17, 60, 23, 55,...
## $ UFE.2   <dbl> 16, 32, 13, 27, 26, 30, 28, 26, 19, 15, 21, 44, 27, 34, 36,...
## $ BPC.2   <dbl> 10, 15, 1, 0, 21, 12, 9, 10, 6, 6, 16, 1, 12, 2, 8, 1, 8, 8...
## $ BPW.2   <dbl> 5, 2, 0, 0, 3, 4, 2, 2, 4, 0, 5, 1, 6, 1, 4, 0, 6, 5, 3, 8,...
## $ NPA.2   <dbl> 23, 46, 19, 22, 44, 33, 11, 38, 11, 23, 50, 21, 61, 27, 26,...
## $ NPW.2   <dbl> 17, 39, 12, 13, 30, 26, 10, 27, 8, 15, 32, 14, 44, 14, 17, ...
## $ TPW.2   <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
## $ ST1.2   <dbl> 6, 6, 2, 2, 6, 6, 3, 6, 6, 6, 7, 4, 6, 7, 7, 6, 6, 6, 3, 6,...
## $ ST2.2   <dbl> 6, 6, 4, 2, 7, 6, 4, 6, 6, 4, 6, 2, 6, 1, 6, 4, 6, 6, 6, 7,...
## $ ST3.2   <dbl> 6, 7, 3, 4, 7, 7, 6, 7, 6, 2, 6, 3, 6, 4, 7, 3, 6, 6, 6, 6,...
## $ ST4.2   <dbl> NA, 7, NA, NA, NA, NA, 3, NA, NA, NA, 6, NA, 6, 3, NA, NA, ...
## $ ST5.2   <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
```

9. Concaténer les tables en ajoutant une variable permettant d'identifier le tournoi. On pourra utiliser `bind_rows()` avec l'option `.id`.

```
tbl <- bind_rows(RolandGarros = rg_tbl,
                 Wimbledon = w_tbl,
```

```
                .id = "Tournoi")
tbl %>%
  group_by(Tournoi) %>%
  summarise(n = n(),
            .groups = 'drop') # Évite un message d'avertissement
```

```
## # A tibble: 2 x 2
##   Tournoi          n
##   <chr>        <int>
## 1 RolandGarros   125
## 2 Wimbledon      114
```

10. Afficher les matchs de Federer pour chaque tournoi.

```
# Aucun match de Federer à Wimbledon ?
tbl %>%
  filter(Player1 == "Roger Federer" | Player2 == "Roger Federer") %>%
  select(Tournoi, Player1, Player2)
```

```
## # A tibble: 5 x 3
##   Tournoi      Player1             Player2
##   <chr>        <chr>               <chr>
## 1 RolandGarros Pablo Carreno-Busta Roger Federer
## 2 RolandGarros Somdev Devvarman    Roger Federer
## 3 RolandGarros Julien Benneteau    Roger Federer
## 4 RolandGarros Gilles Simon        Roger Federer
## 5 RolandGarros Jo-Wilfried Tsonga  Roger Federer
```

```
# Il faut faire attention ...
tbl %>%
  filter(grepl("Federer", Player1) | grepl("Federer", Player2)) %>%
  select(Tournoi, Player1, Player2)
```

```
## # A tibble: 7 x 3
##   Tournoi      Player1             Player2
##   <chr>        <chr>               <chr>
## 1 RolandGarros Pablo Carreno-Busta Roger Federer
## 2 RolandGarros Somdev Devvarman    Roger Federer
## 3 RolandGarros Julien Benneteau    Roger Federer
## 4 RolandGarros Gilles Simon        Roger Federer
## 5 RolandGarros Jo-Wilfried Tsonga  Roger Federer
## 6 Wimbledon    V.Hanescu           R.Federer
## 7 Wimbledon    S.Stakhovsky        R.Federer
```

11. Comparer les nombres d'aces par matchs à chaque tours pour les tournois de Roland Garros et Wimbledon.

```
tbl %>%
  mutate(aces = ACE.1 + ACE.2) %>%
  group_by(Round, Tournoi) %>%
  summarise(mean_aces = mean(aces),
            .groups = 'drop') %>% # Évite un message d'avertissement
  spread(Round, mean_aces)
```

```
## # A tibble: 2 x 8
##   Tournoi        `1`   `2`   `3`   `4`   `5`   `6`   `7`
##   <chr>        <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
```

```
## 1 RolandGarros  13.5  13.2  12.6  9.12   7    10       6
## 2 Wimbledon     21.1  23.9  24   24.4   26.5  27.5    13
```