Exercise ARCHE

This is a subtitle

Name of the Student

But du Workshop:

Le but principal est de comprendre la reproducibilité des résultats dans la recherche scientifique.

> Nous allons reprendre plusieurs éléments de base du langage **R**. Une bonne compréhension des bases du langage, bien qu'un peu ardue de prime abord, permet de comprendre le sens des commandes qu'on utilise et de pleinement exploiter la puissance que **R** offre en matière de manipulation de données.

Exercise I: Bases de la programmation R

Fonctions & Objets

Résumé des elements importants

1. Les vecteurs sont l'un des objets de base de R et correspondent à une liste de valeurs. Leurs propriétés fondamentales sont :
2. Clasess:

- Numerical, Character, Integer or Logical

2. Objects

- Vectors, Lists, Data frames, Factors.

3. Operations

- Subsetting, Logical subsetting

Variables

- Definez une variable type numerique, character, entier et logique.

```
# Définition d'une variable type 'numerique'
#a <- ____
```

```
a_int <- 20
b_int <- 30
```

```
# Définition d'une variable type 'Entier'
#b <- ________
```

```r
a_int <- 20L
b_int <- 30L
```

```r
# Définition d'une variable type 'Logical'
# c <- ______
```

```r
a_log <- TRUE
b_log <- FALSE
```

```r
# Définition d'une variable type 'character' (texte)
#d <- _____
```

```r
a_char <- "Bonjour"
b_char <- "Comment ça va?"
```

Vecteur

La fonction `c()` (‘*combination fonction*’) permet de creer de vecteur:

```r
# Faites un vecteur de longitud 5 numerique.
# vecteur <- ____________
```

- Determinez:
  - Age moyenne des des étudiants dans la variable `moyenne_global`.
  - Longueur du vecteur étudiants. (voir `?length`)

```r
etudiants <- c("Maelle", "Luca", "Sandrine", "Marcelo", "Jean")
taille  <-  c(188, 173, 187, 164, 178)
age <- c(32, 23, 35, 35, 54)


# moyenne_global <- ....

# total_etu <- ...

# Lé résultat de la somme est:  **`r moyenne_global`
# La totalité des étudiantes est: **`r total_etu`
```

List

- Faites une liste avec les vecteurs que vous avez fait dans l’exercise précedente

```r
#list(__________)
```

```r
# Creation d'une List
v1 <- c("Maelle", "Luca", "Sandrine", "Marcelo", "Jean")
v2  <-  c(188, 173, 187, 164, 178)
v3 <- c(32, 23, 35, 35, 54)

a_list <- list(etudiant = v1,
               taille = v2,
               age = v3 )
a_list
```

```
## $etudiant
## [1] "Maelle"   "Luca"     "Sandrine" "Marcelo"  "Jean"
##
```

```
## $taille
## [1] 188 173 187 164 178
##
## $age
## [1] 32 23 35 35 54
```

Data frames

- Pourquoi le data.frame suivant marque un erreur?. Faites la correction necessaire.

```
etudiants <- c("Maelle", "Luca", "Sandrine", "Marcelo", "Jean")
taille  <-  rnorm(8, mean = 173, sd = 5 )

#a_dataframe <-  data.frame(etudiants, taille)
```

- Seleccionez la colonne `Species` du data.frame `iris`, et trouvez les valeurs minimal, maximal, moyenne de la colonne `Sepal.Length`.

```
data <- iris$Sepal.Length
```

Exercise II: Bases de la visualisation de données

##Fonction `ggplot()`

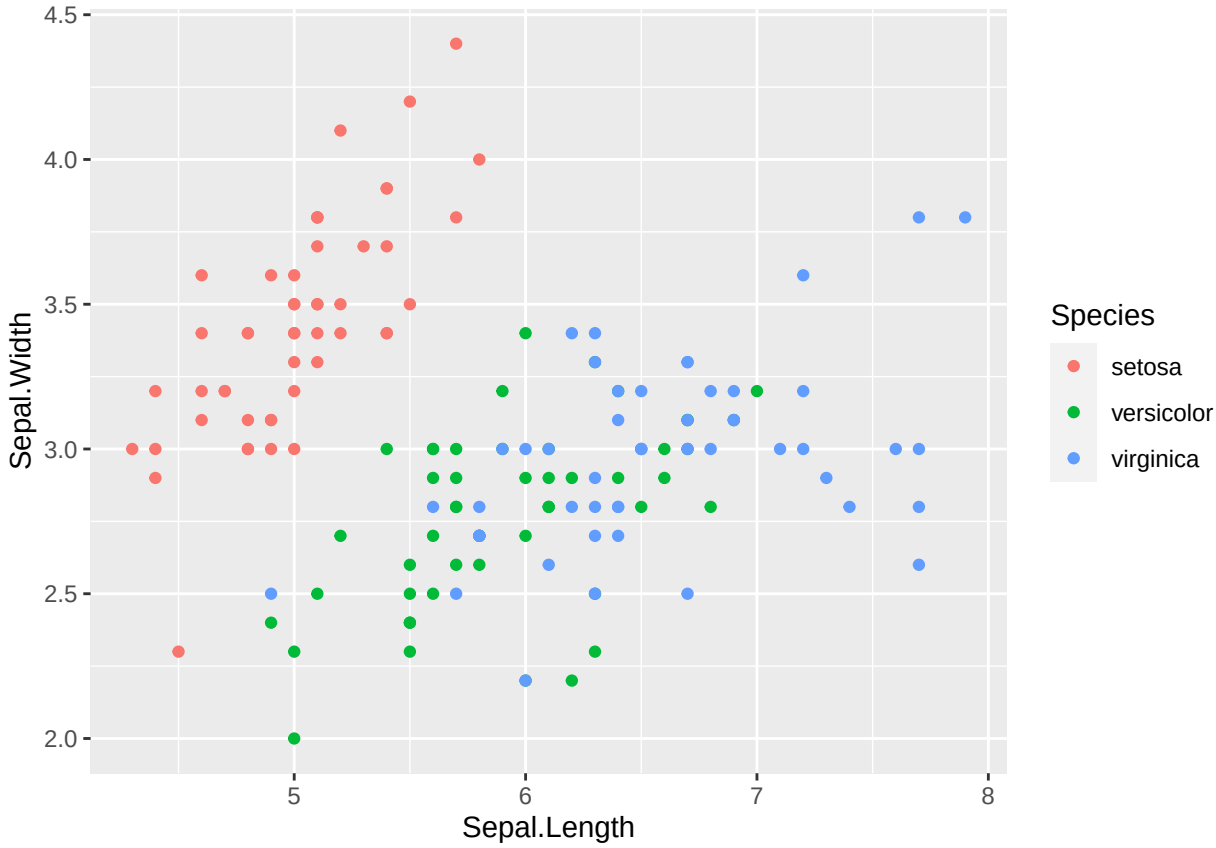
A retenir de ce module:

1. Créer des graphiques avec un **modèle** (ou **template**) {ggplot2} réutilisable
2. Ajouter des variables à un graphique avec **aesthetics**
3. Sélectionner le "type" de votre graphique avec **geoms**

- Faites un nuages de points avec les variables `Sepal.Length` et `Sepal.Width` du data.frame `iris` en faisant la distintion des couleur pour chaque spèce de fleur.

```
# ggplot(data = <DATA>) +
#   aes(x = <X>, y = <Y>)
#  <GEOM_FUNCTION>()
```

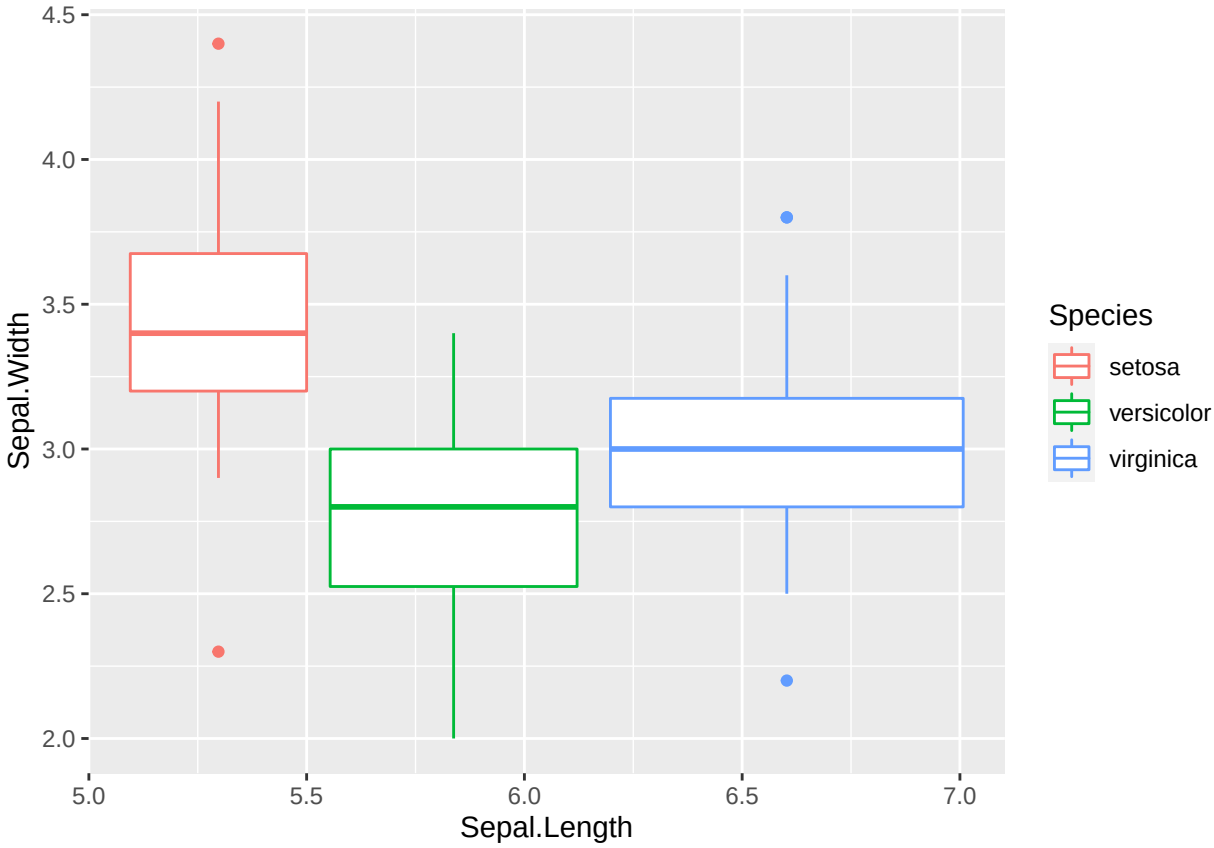
```
ggplot(data = iris) +
  aes(x = Sepal.Length, y = Sepal.Width, color = Species) +
  geom_point()
```



- Faites un boxplot avec memes variables `Sepal.Length` et `Sepal.Width` du data.frame `iris` en faisanat la distintion des coulor pour chaque Spece de fleur.
- Pouvez-vous ajouter les titres, et labels dans le graphique? (regardez `?labs`)

```
# ggplot(data = <DATA>) +
#   aes(x = <X>, y = <Y>)
#  <GEOM_FUNCTION>()
```

```
ggplot(data = iris) +
  aes(x = Sepal.Length, y = Sepal.Width, color = Species) +
  geom_boxplot()
```



Exercise III: Travailler avec les tibbles

Import Data

`readr` et `readxl` permettent d'importer des données tabulaires depuis des fichiers texte ou Excel. Une interface intégrée à RStudio facilite leur usage en permettant de modifier les options d'importation et d'avoir un aperçu en temps réel.

```
# Load the library
library(tidyverse)
```

- Lire une base de donnes CSV:

```
data.csv <- read_csv2("data/nat2020.csv")
```

```
## i Using "',' as decimal and "'.'" as grouping mark. Use `read_delim()` for more control.
```

```
## Rows: 667364 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr (2): preusuel, annais
## dbl (2): sexe, nombre
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
data.csv <- data.csv[100:110 ,]
print(data.csv)
```

```
## # A tibble: 11 x 4
##     sexe preusuel       annais nombre
##    <dbl> <chr>          <chr>   <dbl>
##  1     1 _PRENOMS_RARES 1999    11738
##  2     1 _PRENOMS_RARES 2000    12693
##  3     1 _PRENOMS_RARES 2001    13406
##  4     1 _PRENOMS_RARES 2002    14529
##  5     1 _PRENOMS_RARES 2003    15426
##  6     1 _PRENOMS_RARES 2004    16568
##  7     1 _PRENOMS_RARES 2005    17491
##  8     1 _PRENOMS_RARES 2006    18435
##  9     1 _PRENOMS_RARES 2007    18878
## 10     1 _PRENOMS_RARES 2008    19278
## 11     1 _PRENOMS_RARES 2009    20031
```

- Lire un fichier EXCEL

```
# Lire un base de donnes Excel
library(readxl)

# Don't show up
data.excel <- read_excel("data/Test_Attrakdiff.xls" )

#identifying the onglets
onglets <- excel_sheets("data/Test_Attrakdiff.xls")

# Reading the onglet that corresponds
data.excel <- read_excel("data/Test_Attrakdiff.xls",  sheet = onglets[2])
```

```
## New names:
## * `` -> ...2
## * `` -> ...3
## * `` -> ...4
## * `` -> ...5
## * `` -> ...6
## * ...
```

```
# Reading the onglet that corresponds
data.excel <- read_excel("data/Test_Attrakdiff.xls",
                         sheet = onglets[2],
                         skip = 2)
```

```
## New names:
## * `` -> ...1
## * `` -> ...30
## * `` -> ...31
## * `` -> ...32
```

```
# Selecting the corresponding rows
data.excel <- data.excel %>% slice(1:11)

# Changing names columns
col_names <- names(data.excel)
col_names[1] <- c("Participant")
names(data.excel) <- col_names

data.excel <-
```

```
  data.excel %>%  select(Participant : QP7)
```

Export de tableaux de données

On peut avoir besoin d'exporter un tableau de données dans R vers un fichier dans différents formats. La plupart des fonctions d'import disposent d'un équivalent permettant l'export de données. On citera notamment :

- `write_csv`, `write_csv2` permettent d'enregistrer un data frame ou un tibble dans un fichier au format texte délimité

```
# Exporté des donnes

# With csv et csv2
write_csv(data.csv, file= "data/export-csv.csv")
write_csv2(data.csv, file= "data/export-csv2.csv")
```

Tibble vs Data.Frame

Qu'est-ce qu'un tibble ?

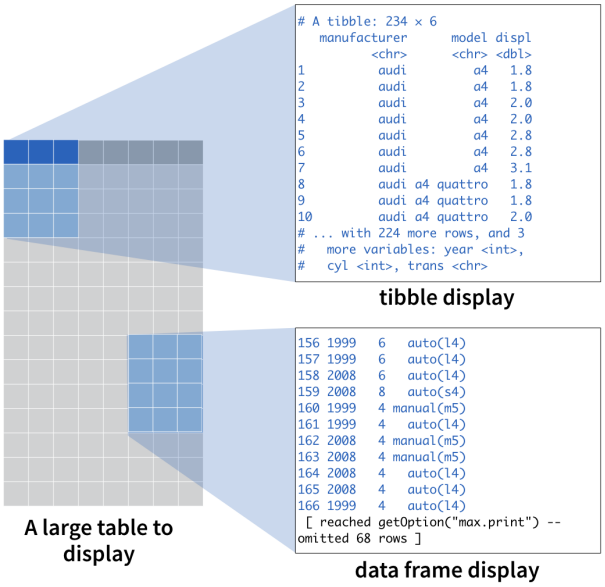
Un tibble est un type spécial de table. R affiche les tibbles de manière astucieuse chaque fois que le package **tibble** est chargé.

- R n'affichera alors uniquement que les dix premières lignes d'un tibble ainsi que toutes les colonnes qui tiennent dans la fenêtre de votre console.
- R ajoute également des informations récapitulatives utiles sur les composants tibble, telles que les types de données de chaque colonne et la taille totale du jeu de données.

Chaque fois que vous n'avez pas le package `{tibble}` chargé, R affichera le tibble comme s'il s'agissait d'un `data.frame`. En fait, les tibbles **sont** des **data.frames**, mais dans une version améliorée.

Vous pouvez penser à la différence entre l'affichage du data.frame et l'affichage du tibble comme ceci :

```
knitr::include_graphics("figures/tibble_display.png")
```



```
data.frame(data.csv)
```

```
##    sexe       preusuel annais nombre
## 1     1 _PRENOMS_RARES   1999  11738
## 2     1 _PRENOMS_RARES   2000  12693
## 3     1 _PRENOMS_RARES   2001  13406
## 4     1 _PRENOMS_RARES   2002  14529
## 5     1 _PRENOMS_RARES   2003  15426
## 6     1 _PRENOMS_RARES   2004  16568
## 7     1 _PRENOMS_RARES   2005  17491
## 8     1 _PRENOMS_RARES   2006  18435
## 9     1 _PRENOMS_RARES   2007  18878
## 10    1 _PRENOMS_RARES   2008  19278
## 11    1 _PRENOMS_RARES   2009  20031
```

```
tibble(data.csv)
```

```
## # A tibble: 11 x 4
##     sexe preusuel       annais nombre
##    <dbl> <chr>          <chr>   <dbl>
##  1     1 _PRENOMS_RARES 1999    11738
##  2     1 _PRENOMS_RARES 2000    12693
##  3     1 _PRENOMS_RARES 2001    13406
##  4     1 _PRENOMS_RARES 2002    14529
##  5     1 _PRENOMS_RARES 2003    15426
##  6     1 _PRENOMS_RARES 2004    16568
##  7     1 _PRENOMS_RARES 2005    17491
##  8     1 _PRENOMS_RARES 2006    18435
##  9     1 _PRENOMS_RARES 2007    18878
## 10     1 _PRENOMS_RARES 2008    19278
## 11     1 _PRENOMS_RARES 2009    20031
```

```
names(data.csv)
```

```
## [1] "sexe"     "preusuel" "annais"   "nombre"
```

```
length(data.csv$`Journal identity`)
```

```
## Warning: Unknown or uninitialised column: `Journal identity`.
```

```
## [1] 0
```

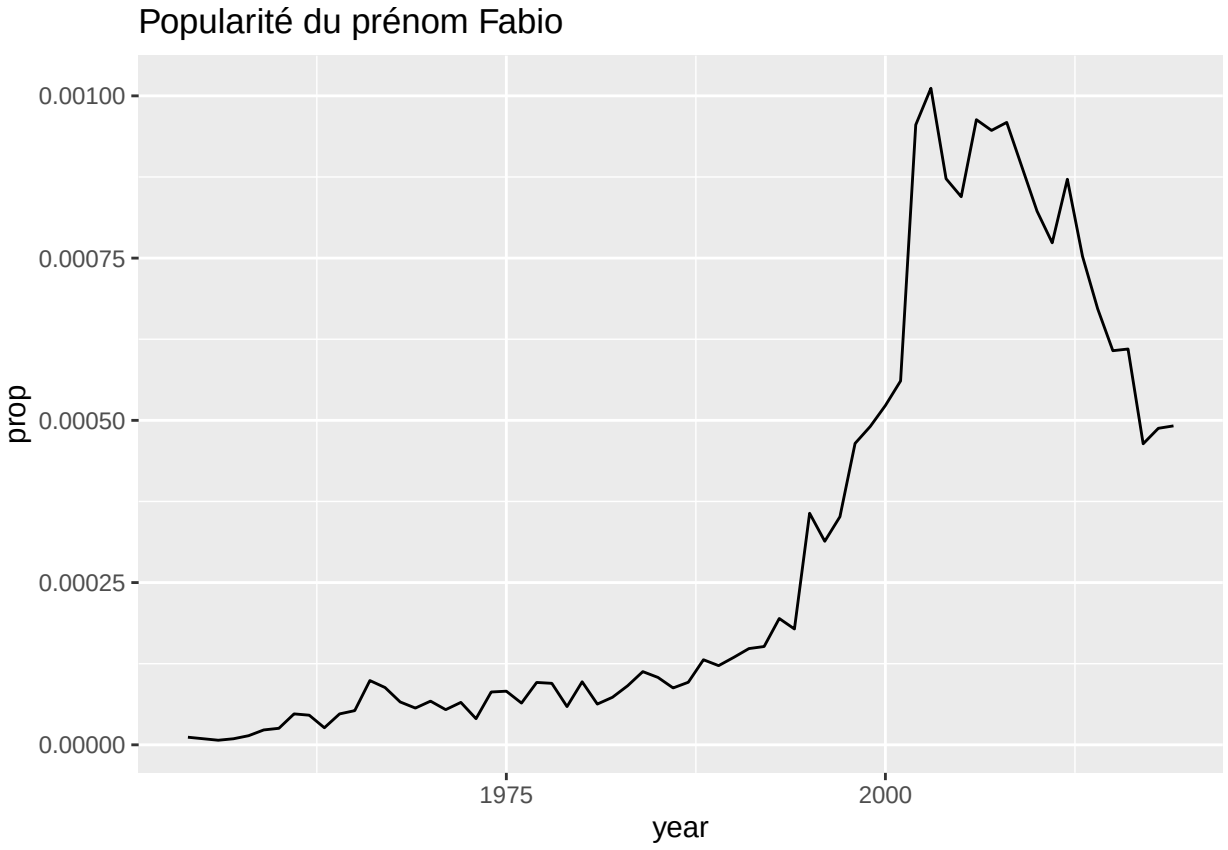
```
# View tibles
#View(data.csv)
```

Exercise IV: Extraire et retirer de l'information avec {dplyr}

Appliquez vos connaissances de {dplyr} pour relever les challenges suivants.

Challenge de votre Prenom

- Graphiquez la curve de votre prenom.
- Dans quelle année specifiquement la popularité de votre prenom a été maximale?




```
## [1] 0.001011709
```

```
## # A tibble: 1 x 5
##    year sex   name      n    prop
##   <dbl> <chr> <chr> <int>   <dbl>
## 1  2003 M     Fabio   389 0.00101
```

Challenge du Top 10

Identifiez les Top 10 prenoms utilisez en France?

```
#top_10 <-
# prenoms_france %>% __________ %>%
```

```
tops <-
  prenoms_france %>%
  group_by(name, sex) %>%
  summarise(total = sum(n)) %>%
  ungroup() %>%
  top_n(10, total)
```

```
## `summarise()` has grouped output by 'name'. You can override using the `.groups`
## argument.
```

Challenge "number one" - focus sur les garçons

Combien de prénoms de garçons distincts ont atteint le rang de numéro 1 au cours d'une année ?

```
# prenoms_france %>%
# __________ %>% ....
```

```
prenoms_france %>%
  group_by(year, sex) %>%
  mutate(rank = min_rank(desc(n))) %>%
  filter(rank == 1, sex == "M") %>%
  ungroup() %>%
  summarise(distinct = n_distinct(name))
```

```
## # A tibble: 1 x 1
##   distinct
##      <int>
## 1       14
```

Challenge "number one" - focus sur les filles

Combien de prénoms de filles distincts ont atteint le rang de numéro 1 au cours d'une année ?

```
# prenoms_france %>%
# __________ %>% ....
```

```
prenoms_france %>%
  group_by(year, sex) %>%
  mutate(rank = min_rank(desc(n))) %>%
  filter(rank == 1, sex == "F") %>%
  ungroup() %>%
  summarise(distinct = n_distinct(name))
```

```
## # A tibble: 1 x 1
##   distinct
##      <int>
## 1       19
```

Exemple Article Live: Attractif

Tidy

Les principes d'un jeu de données tidy sont les suivants :

- chaque variable est une colonne
- chaque observation est une ligne
- chaque type d'observation est dans une table différente

Final Exercise

Gapminder challenge

- Inspiré par Hans Rosling

- Grafiquez et compare l'espérance de vie moyenne sur l'ensemble des années pour la France et u autre pays?

```
library(gapminder)
gapminder
```

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # ... with 1,694 more rows
```

```
gapminder %>%
  filter(country %in% c("France", "Afghanistan")) %>%
ggplot(aes(x = year, y = lifeExp, color=country)) +
  geom_line()
```