

# Objectifs

- Recombiner des tableaux larges en longs
- Recombiner des tableaux longs en larges
- Séparer et unifier des colonnes dans un tableau de données

# Les fonctions svTidy/Tidyverse

L'instruction `SciViews::R` ou `SciViews::R()` qui est équivalente, est employée pour charger une série cohérente de packages R. Nous l'utilisons en début de script R ou dans le premier chunk d'un document Quarto ou R Markdown.

```
SciViews::R()
```

Vous employez des tableaux de données depuis le premier module de ce cours pour réaliser des graphiques. Vous allez à présent apprendre à les manipuler. Il existe dans R plusieurs types d'objets pour contenir des jeux de données, dont **data.frame**, tibble (objet **tbl\_df**) ou **data.table**. Par défaut dans `SciViews::R`, vous travaillez avec des **data.frames**. La vitesse de traitements et la gestion en mémoire des objets **data.frame** en font le choix idéal pour traiter des gros jeux de données. Actuellement, vous n'avez pas encore traité de gros jeux de données mais cela va venir plus vite que vous ne l'imaginez.

Le remaniement de données est tout un art que vous devez maîtriser. Vous devez être capable de sélectionner des colonnes, de filtrer des lignes, de trier un tableau, de résumer vos observations...

Parmi les fonctions de remaniement de tableaux, nous avons les fonctions svTidy qui visent à accélérer le traitement dans R. Ces fonctions sont compatibles avec les objets **data.frame**. Elles sont suffixées avec un “\_” pour les reconnaître facilement.

Un second ensemble est constitué des fonctions Tidyverse (<https://www.tidyverse.org>), et portant le même nom que leurs homologues svTidy, mais non suffixées d'un “\_”. Elles sont couramment utilisées dans R, mais, dans la plupart des cas, plus lentes que les fonctions svTidy correspondantes, et elles n'utilisent pas l'interface formule qui vous est familière avec, entre autres, `chart()`.

Nous retrouvons donc des paires de fonctions svTidy/Tidyverse qui réalisent des traitements très similaires et ont une syntaxe relativement proche. Elles permettent de :

- sélectionner des colonnes d'un jeu de données avec `select_()` / `select()`
- filtrer des lignes dans un jeu de données avec `filter_()` / `filter()`
- calculer de nouvelles variables dans un jeu de données avec `mutate_()` / `mutate()`
- indiquer les variables de regroupement à utiliser dans un tableau avec `group_by_()` / `group_by()`
- résumer les variables d'un jeu de données avec `summarise_()` / `summarise()`

*Nous vous conseillons d'employer préférentiellement les fonctions svTidy. Cependant, dans ce tutoriel nous utiliserons alternativement les deux pour se familiariser avec les deux syntaxes.*

# Deux variétés de crabes

Le tableau de données que vous allez employer porte sur deux variétés de crabes.

```
SciViews::R(lang = "fr")
# Importation des données crabs
(crabs <- read("crabs", package = "MASS"))
```

species <fct>	sex <fct>	index <int>	front <dbl>	rear <dbl>	length <dbl>	width <dbl>	depth <dbl>					
B	M	1	8.1	6.7	16.1	19.0	7.0					
B	M	2	8.8	7.7	18.1	20.8	7.4					
B	M	3	9.2	7.8	19.0	22.4	7.7					
B	M	4	9.6	7.9	20.1	23.1	8.2					
B	M	5	9.8	8.0	20.3	23.0	8.2					
B	M	6	10.8	9.0	23.0	26.5	9.8					
B	M	7	11.1	9.9	23.8	27.1	9.8					
B	M	8	11.6	9.1	24.5	28.4	10.4					
B	M	9	11.8	9.6	24.2	27.8	9.7					
B	M	10	11.8	10.5	25.2	29.3	10.3					
1-10 of 200 rows			Previous	1	2	3	4	5	6	...	20	Next

Ce tableau contient 200 observations et 8 variables. Les carapaces d'individus mâles et femelles de deux variétés de crabes ont été mesurées sous toutes les coutures.

## Calculer de nouvelles variables

Calculez sur le tableau `crabs` à l'aide de la fonction `svTidy::mutate_()` :

- le logarithme ( `log()` ) de la longueur de la carapace ( `length` ) des crabes et nommez cette nouvelle variables `log_length`
- la racine carrée ( `sqrt()` ) de la largeur de la carapace ( `width` ) et nommez cette nouvelle variable `sqrt_width`
- Divisez la variable lobe frontal ( `front` ) par 1000 et nommez cette nouvelle variable `front_m`

Affichez ensuite les premières lignes du tableau ( `head()` ). Par défaut, les six premières lignes sont affichées. Vous devez obtenir le tableau ci-dessous :

species <fct>	s... <fct>	index <int>	front <dbl>	rear <dbl>	length <dbl>	width <dbl>	depth <dbl>	log_length <dbl>	sqrt_width <dbl>
B	M	1	8.1	6.7	16.1	19.0	7.0	2.778819	4.358899
B	M	2	8.8	7.7	18.1	20.8	7.4	2.895912	4.560702
B	M	3	9.2	7.8	19.0	22.4	7.7	2.944439	4.732864
B	M	4	9.6	7.9	20.1	23.1	8.2	3.000720	4.806246
B	M	5	9.8	8.0	20.3	23.0	8.2	3.010621	4.795832
B	M	6	10.8	9.0	23.0	26.5	9.8	3.135494	5.147815

6 rows | 1-10 of 11 columns

Code R   ↺ Start Over   💡 Hints   ▶ Run Code   ☒ Submit Answer

```
1 ____ <- ____(____,  
2   ____ = ____,  
3   ____ = ____,  
4   ____ = ____)  
5 # Visualisation des premières lignes du tableau  
6 ____(____)
```

Continue

---

# Conclusion

Bravo ! Vous venez de terminer votre séance d'exercices relative à la manipulation des données.

Vous avez maintenant un aperçu de :

- notions relatives au remaniement des données avec les fonctions `select_()` / `select()` ,  
`filter_()` / `filter()` , `mutate_()` / `mutate()` , `group_by_()` / `group_by()` ,  
`summarise_()` / `summarise()`
- chaînages des instructions dans un pipeline avec `%>.%`

**Laissez-nous vos impressions sur ce learnr**

Entrez vos commentaires ici...

Submit Answer