Qu'est ce que DPLYR ?

Marié Théo

Décembre 2020

Dans le Data Management, la manipulation et le traitement des bases de données est primordiale pour un Data Analyst. Nous nous penchons ici sur un package qui va nous faciliter la vie avec cette tâche : **le package dplyr**. Vous n'aurez alors plus aucun soucis pour travailler et modifier des tableaux de données (data.fram ou tibble).

Pour utiliser ce package, il faut tout d'abord l'installer, pour qu'il soit sauvegardé dans votre pc. Pour se faire on peut soit installer tidyverse, et dplyr.

install.packages("tidyverse")

install.packages("dplyr")

Ici, pour vous montrer ce package, nous devons utiliser une base de données. Nous allons prendre la fameuse base de donnée **titanic**. Aussi, nous nous focaliserons sur la table **titanic_gender_class_model**. Pour se faire, il faut simplement télécharger la base de donnée titanic puis indiquer que l'on invoquera la table titanic_gender_class_model.

install.packages("titanic")

```
library(titanic)
data("titanic_gender_class_model")
```

Nous avons enfin notre base de données ! Maintenant, il faut maintenant pouvoir plonger dedans. Pour cela, voici quelques fonctions basiques proposées dans le package qui vont nous permettre de nous balader dans cette base de données.

Imaginons que vous voulez vous occuper des valeurs contenues dans le tableau, pour faire des sommes, des tris etc...

La fonction idéale pour savoir la valeur minimum dans une colonne est **LA FONCTIO MIN**.

```
data("titanic_gender_class_model")
min(titanic_gender_class_model)
```

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

Exactement pareil quand on cherche le maximum, **LA FONCTION MAX**

```
data("titanic_gender_class_model")
max(titanic_gender_class_model)
```

```
## [1] 1309
```

Pour finir, nous allons voir **LA FONCTION SLICE** Elle permet de sélectionner des LIGNES du tableau, que ce soit une en particulier, la première, la dernière, ou même une aléatoirement.

Nous pouons ici sélectionner la première ligne du tableau grâce à **slice_head**

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
data("titanic_gender_class_model")
slice_head(titanic_gender_class_model)
```

```
##   PassengerId Survived
## 1         892        0
```

Imaginons que nous voulons prendre une ligne aléatoire pour vérifier si des résultats concordent avec nos suppositions : **slice_sample**

```
library(dplyr)
data("titanic_gender_class_model")
slice_sample(titanic_gender_class_model)
```

```
##   PassengerId Survived
## 1         926        0
```

Sur notre table, ous sommes curieux de savoir ce qui se passe sur la case 666 : **slice(LaTable, 666)** (n'ayant pas 666 Lignes, on prendra la 333)

```
library(dplyr)
data("titanic_gender_class_model")
slice(titanic_gender_class_model, 333)
```

```
##   PassengerId Survived
## 1        1224        0
```

Cas spécial : nous voulons prendre un interval de lignes :**slice(LaTable, 4:9)**

```
library(dplyr)
data("titanic_gender_class_model")
slice(titanic_gender_class_model, 4:9)
```

```
##   PassengerId Survived
## 1         895        0
## 2         896        1
## 3         897        0
## 4         898        1
## 5         899        0
## 6         900        1
```

Vous pouvez trouver beaucoup de ces fonctions sur ce **GITHUB**