Le travail de Jean Souris est disponible ici

BENOUCIEF Amine

22/12/2020

SYNTHESE DU TRAVAIL EN QUESTION

dplyr est une extension facilitant le traitement et la manipulation de données contenues dans une ou plusieurs tables. Elle propose une syntaxe claire et cohérente, sous formes de verbes, pour la plupart des opérations de ce type. Jean a reussi a bien explique les fonctionnalites de dplyr et n'a pas hesite a commente clairement chaque ligne de son code afin de faciliter la lecture.

Introduction :

Tout d'abord, il faut installer le package dplyr pour cette demonstration :

```
#install.packages("dplyr")
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
```

Dplyr sert à la manipulation de bases de données sous forme de tableau, donc pouvoir les réarranger, les filtrer, les trier, plein beaucoup d'autres fonctions.

Mais, avant de pouvoir modifier une base de donnée, il faut en selectionner une ; c'est pour cela que nous allons utiliser les fonctions ci-dessous :

```
#install.packages("nycflights13")
library("nycflights13")

## Warning: package 'nycflights13' was built under R version 4.0.3
```

Après avoir installé le package contenant notre base de donnée, nous allons seelctionner 2 tableaux que nous utiliserons au cours de cette démonstration :

```
data(flights)
data(airports)
```

Dans cette partie, nous allons voir 3 principaux verbes que nous pouvons utiliser sur dplyr.

Slice

Le premier verbe que nous allons voir est “slice” et permet globalement de selectionner à notre guise différentes lignes d’un tableau afin de les afficher :

Nous allons afficher une certaine ligne de la collone “airlines” et voir ce qui s’affiche :

```
slice(airports, 537)
```

```
## # A tibble: 1 x 8
##   faa   name              lat   lon   alt    tz dst   tzone
##   <chr> <chr>           <dbl> <dbl> <dbl> <dbl> <chr> <chr>
## 1 GGG   East Texas Rgnl  32.4 -94.7   365    -6 A     America/Chicago
```

Nous avons donc accès à plusieurs informations d’un aéroport précis, tel que ses coordonnées géographiques, à savoir lattitude, longitude, même altitude mais aussi à son nom raccourcis et sa zone géographique.

La fonction slice nous permet également de selectionner plusieurs lignes à la fois en utilisant un interval :

```
slice(airports, 9:27)
```

```
## # A tibble: 19 x 8
##    faa   name                       lat    lon   alt    tz dst   tzone
##    <chr> <chr>                    <dbl>  <dbl> <dbl> <dbl> <chr> <chr>
##  1 0P2   Shoestring Aviation Airf~  39.8  -76.6  1000    -5 U
America/New_Y~
##  2 0S9   Jefferson County Intl      48.1 -123.    108    -8 A
America/Los_A~
##  3 0W3   Harford County Airport     39.6  -76.2   409    -5 A
America/New_Y~
##  4 10C   Galt Field Airport         42.4  -88.4   875    -6 U
America/Chica~
##  5 17G   Port Bucyrus-Crawford Co~  40.8  -83.0  1003    -5 A
America/New_Y~
##  6 19A   Jackson County Airport     34.2  -83.6   951    -5 U
America/New_Y~
##  7 1A3   Martin Campbell Field Ai~  35.0  -84.3  1789    -5 A
America/New_Y~
##  8 1B9   Mansfield Municipal        42.0  -71.2   122    -5 A
America/New_Y~
##  9 1C9   Frazier Lake Airpark       54.0 -125.    152    -8 A
America/Vanco~
## 10 1CS   Clow International Airpo~  41.7  -88.1   670    -6 U
America/Chica~
## 11 1G3   Kent State Airport         41.2  -81.4  1134    -5 A
America/New_Y~
```

```
## 12 1G4   Grand Canyon West Airport  35.9 -114.   4813    -7 A
America/Phoen~
## 13 1H2   Effingham Memorial Airpo~  39.1  -88.5   585    -6 A
America/Chica~
## 14 1OH   Fortman Airport            40.6  -84.4   885    -5 U
America/New_Y~
## 15 1RL   Point Roberts Airpark      49.0 -123.     10    -8 A
America/Los_A~
## 16 23M   Clarke CO                  32.1  -88.4   320    -6 A
America/Chica~
## 17 24C   Lowell City Airport        43.0  -85.3   681    -5 A
America/New_Y~
## 18 24J   Suwannee County Airport    30.3  -83.0   104    -5 A
America/New_Y~
## 19 25D   Forest Lake Airport        45.2  -93.0   925    -6 A
America/Chica~
```

Ici, nous avons selectionné les lignes 9 à 27 du tableau de données des aéroports.

Hormis la selection de lignes au choix d'un tableau, la fonction slice nous permet également d'en selectionner de manière aléatoire grâce au verbe "slice_sample" :

```
airports %>% slice_sample(n=6)
```

```
## # A tibble: 6 x 8
##   faa   name                          lat    lon   alt    tz dst   tzone
##   <chr> <chr>                       <dbl>  <dbl> <dbl> <dbl> <chr> <chr>
## 1 GBN   Great Bend Municipal         38.3  -98.9  1887    -6 A
America/Chi~
## 2 LRO   Mount Pleasant Regional-Fai~  32.5  -79.5    12    -5 A     <NA>
## 3 MRB   Eastern WV Regional Airport   39.2  -77.6   554    -5 A
America/New~
## 4 CAR   Caribou Muni                 46.9  -68.0   626    -5 A
America/New~
## 5 EKI   Elkhart Municipal            41.7  -86.0   778    -5 A
America/New~
## 6 GNT   Grants Milan Muni            35.2 -108.   6537    -7 A
America/Den~
```

```
slice(airports, 1:6)
```

```
## # A tibble: 6 x 8
##   faa   name                          lat   lon   alt    tz dst   tzone
##   <chr> <chr>                       <dbl> <dbl> <dbl> <dbl> <chr> <chr>
## 1 04G   Lansdowne Airport            41.1 -80.6  1044    -5 A
America/New_Y~
## 2 06A   Moton Field Municipal Airp~  32.5 -85.7   264    -6 A
America/Chica~
## 3 06C   Schaumburg Regional          42.0 -88.1   801    -6 A
America/Chica~
## 4 06N   Randall Airport              41.4 -74.4   523    -5 A
```

```
America/New_Y~
## 5 09J   Jekyll Island Airport        31.1 -81.4    11    -5 A
America/New_Y~
## 6 0A9   Elizabethton Municipal Air~  36.4 -82.2  1593    -5 A
America/New_Y~
```

Comme vous pouvez le constater, le premier tableau a généré aléatoirement 6 lignes du tableua aéroport, lorsque le second a slectionné les 6 premières.

NB : Nous pouvons également tirer des lignes du tableau en partant du bas ou du haut grâce aux verbes “slice_head” et “slice_tail” :

```
airports %>% slice_head(n=3)

## # A tibble: 3 x 8
##   faa   name                          lat   lon   alt    tz dst   tzone
##   <chr> <chr>                       <dbl> <dbl> <dbl> <dbl> <chr> <chr>
## 1 04G   Lansdowne Airport            41.1 -80.6  1044    -5 A
America/New_Y~
## 2 06A   Moton Field Municipal Airp~  32.5 -85.7   264    -6 A
America/Chica~
## 3 06C   Schaumburg Regional          42.0 -88.1   801    -6 A
America/Chica~

airports %>% slice_tail(n=3)

## # A tibble: 3 x 8
##   faa   name                         lat   lon   alt    tz dst   tzone
##   <chr> <chr>                      <dbl> <dbl> <dbl> <dbl> <chr> <chr>
## 1 ZWI   Wilmington Amtrak Station  39.7 -75.6     0    -5 A
America/New_York
## 2 ZWU   Washington Union Station   38.9 -77.0    76    -5 A
America/New_York
## 3 ZYP   Penn Station               40.8 -74.0    35    -5 A
America/New_York
```

De même, nous pouvons tirer au hasard 5% de lignes de notre tableau en utilisant la fonction “prop” tel que :

```
airports %>% slice_sample(prop = 0.05)

## # A tibble: 72 x 8
##    faa   name                        lat    lon   alt    tz dst   tzone
##    <chr> <chr>                     <dbl>  <dbl> <dbl> <dbl> <chr> <chr>
##  1 PSP   Palm Springs Intl         33.8 -117.    477    -8 A
America/Los_An~
##  2 HUT   Hutchinson Municipal Ai~  38.1  -97.9  1543    -6 A
America/Chicago
##  3 FOE   Forbes Fld                39.0  -95.7  1078    -6 A
America/Chicago
##  4 ZSF   Springfield Amtrak Stat~  42.1  -72.6    65    -5 A
America/New_Yo~
```

```
##  5 VNY   Van Nuys                  34.2 -118.    802    -8 A
America/Los_An~
##  6 LNS   Lancaster Airport         40.1  -76.3   403    -5 A
America/New_Yo~
##  7 ONT   Ontario Intl              34.1 -118.    944    -8 A
America/Los_An~
##  8 DAB   Daytona Beach Intl        29.2  -81.1    34    -5 A
America/New_Yo~
##  9 JKA   Jack Edwards Airport      30.3  -87.7    17    -6 A
America/Chicago
## 10 ADW   Andrews Afb               38.8  -76.9   280    -5 A
America/New_Yo~
## # ... with 62 more rows
```

Il y a également des verbes tels que “slice_min” et “slice_max” qui prennent en compte un argument supplémentaire du tableua choisi afin de filtrer son choix. Par exemple, si je souhaite connaitre les 7 aéroports étant le plus bas, donc ayant la plus faible altitude, j’utilise la fonction suivante :

```
airports %>% slice_max(alt, n=7)
```

```
## # A tibble: 7 x 8
##   faa   name                          lat   lon   alt    tz dst   tzone
##   <chr> <chr>                       <dbl> <dbl> <dbl> <dbl> <chr> <chr>
## 1 TEX   Telluride                    38.0 -108.  9078    -7 A
America/Denver
## 2 TVL   Lake Tahoe Airport           38.9 -120.  8544    -8 A
America/Los_An~
## 3 ASE   Aspen Pitkin County Sardy~  39.2 -107.  7820    -7 A
America/Denver
## 4 GUC   Gunnison - Crested Butte     38.5 -107.  7678    -7 A
America/Denver
## 5 BCE   Bryce Canyon                 37.7 -112.  7590    -7 A
America/Denver
## 6 ALS   San Luis Valley Regional ~   37.4 -106.  7539    -7 A
America/Denver
## 7 LAR   Laramie Regional Airport     41.3 -106.  7284    -7 A
America/Denver
```

```
summary(flights)
```

```
##       year          month             day           dep_time
sched_dep_time
##  Min.   :2013   Min.   : 1.000   Min.   : 1.00   Min.   :   1   Min.   :
106
##  1st Qu.:2013   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 907   1st Qu.:
906
##  Median :2013   Median : 7.000   Median :16.00   Median :1401   Median
:1359
##  Mean   :2013   Mean   : 6.549   Mean   :15.71   Mean   :1349   Mean
:1344
```

```
##  3rd Qu.:2013   3rd Qu.:10.000   3rd Qu.:23.00   3rd Qu.:1744   3rd
Qu.:1729
##  Max.   :2013   Max.   :12.000   Max.   :31.00   Max.   :2400   Max.
:2359
##                                                  NA's   :8255
##    dep_delay          arr_time    sched_arr_time   arr_delay
##  Min.   : -43.00   Min.   :   1   Min.   :   1   Min.   : -86.000
##  1st Qu.:  -5.00   1st Qu.:1104   1st Qu.:1124   1st Qu.: -17.000
##  Median :  -2.00   Median :1535   Median :1556   Median :  -5.000
##  Mean   :  12.64   Mean   :1502   Mean   :1536   Mean   :   6.895
##  3rd Qu.:  11.00   3rd Qu.:1940   3rd Qu.:1945   3rd Qu.:  14.000
##  Max.   :1301.00   Max.   :2400   Max.   :2359   Max.   :1272.000
##  NA's   :8255      NA's   :8713                  NA's   :9430
##    carrier              flight       tailnum             origin
##  Length:336776      Min.   :   1   Length:336776      Length:336776
##  Class :character   1st Qu.: 553   Class :character   Class :character
##  Mode  :character   Median :1496   Mode  :character   Mode  :character
##                     Mean   :1972
##                     3rd Qu.:3465
##                     Max.   :8500
##
##      dest              air_time        distance         hour
##  Length:336776      Min.   : 20.0   Min.   :  17   Min.   : 1.00
##  Class :character   1st Qu.: 82.0   1st Qu.: 502   1st Qu.: 9.00
##  Mode  :character   Median :129.0   Median : 872   Median :13.00
##                     Mean   :150.7   Mean   :1040   Mean   :13.18
##                     3rd Qu.:192.0   3rd Qu.:1389   3rd Qu.:17.00
##                     Max.   :695.0   Max.   :4983   Max.   :23.00
##                     NA's   :9430
##      minute        time_hour
##  Min.   : 0.00   Min.   :2013-01-01 05:00:00
##  1st Qu.: 8.00   1st Qu.:2013-04-04 13:00:00
##  Median :29.00   Median :2013-07-03 10:00:00
##  Mean   :26.23   Mean   :2013-07-03 05:22:54
##  3rd Qu.:44.00   3rd Qu.:2013-10-01 07:00:00
##  Max.   :59.00   Max.   :2013-12-31 23:00:00
##
```

De même si je souhaite connaitres les 10 vols les plus courts effectués en 2013 :

```
flights %>% slice_min(distance, n=10)
```

```
## # A tibble: 50 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time
sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>
<int>
##  1  2013     7    27       NA            106        NA       NA
245
##  2  2013     1     3     2127           2129        -2     2222
2224
```

```
##  3  2013     1     4     1240           1200         40     1333
1306
##  4  2013     1     4     1829           1615        134     1937
1721
##  5  2013     1     4     2128           2129         -1     2218
2224
##  6  2013     1     5     1155           1200         -5     1241
1306
##  7  2013     1     6     2125           2129         -4     2224
2224
##  8  2013     1     7     2124           2129         -5     2212
2224
##  9  2013     1     8     2127           2130         -3     2304
2225
## 10  2013     1     9     2126           2129         -3     2217
2224
## # ... with 40 more rows, and 11 more variables: arr_delay <dbl>, carrier
<chr>,
## #   flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
## #   distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

Select :

Le second verbe que nous allons utiliser s'intitule "select", et, comme son nom l'indique nous permet de selectionner des données d'un tableau et plus précisement des collones de celui-ci tel que :

```
select(flights, origin, time_hour)
```

```
## # A tibble: 336,776 x 2
##    origin time_hour
##    <chr>  <dttm>
##  1 EWR    2013-01-01 05:00:00
##  2 LGA    2013-01-01 05:00:00
##  3 JFK    2013-01-01 05:00:00
##  4 JFK    2013-01-01 05:00:00
##  5 LGA    2013-01-01 06:00:00
##  6 EWR    2013-01-01 05:00:00
##  7 EWR    2013-01-01 06:00:00
##  8 LGA    2013-01-01 06:00:00
##  9 JFK    2013-01-01 06:00:00
## 10 LGA    2013-01-01 06:00:00
## # ... with 336,766 more rows
```

Ici nous avons donc les collones nous indiquant l'origine et l'heure de nos vols.

Nommer toutes les colonnes peut paraître rébarbatif, nous pouvons donc selectionner un interval contenant les colonnes que nous souhaitons tel que :

```
select(flights, dep_time:dep_delay)
```

```
## # A tibble: 336,776 x 3
##    dep_time sched_dep_time dep_delay
##       <int>          <int>     <dbl>
##  1      517            515         2
##  2      533            529         4
##  3      542            540         2
##  4      544            545        -1
##  5      554            600        -6
##  6      554            558        -4
##  7      555            600        -5
##  8      557            600        -3
##  9      557            600        -3
## 10      558            600        -2
## # ... with 336,766 more rows
```

Les colonnes situées entre “dep_time” et dep“delay” comprises sont donc affichées.

En revanche, si, avant le nom de chaque colonne nous faisons apparaître le symbole “-”, alors le tableau s’affiche entièrement en ayant soustrait les colonnes selectionnées :

```
select(flights, -origin, -time_hour)
```

```
## # A tibble: 336,776 x 17
##     year month   day dep_time sched_dep_time dep_delay arr_time
sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>
<int>
##  1  2013     1     1      517            515         2      830
819
##  2  2013     1     1      533            529         4      850
830
##  3  2013     1     1      542            540         2      923
850
##  4  2013     1     1      544            545        -1     1004
1022
##  5  2013     1     1      554            600        -6      812
837
##  6  2013     1     1      554            558        -4      740
728
##  7  2013     1     1      555            600        -5      913
854
##  8  2013     1     1      557            600        -3      709
723
##  9  2013     1     1      557            600        -3      838
846
## 10  2013     1     1      558            600        -2      753
745
## # ... with 336,766 more rows, and 9 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, dest <chr>, air_time
<dbl>,
## #   distance <dbl>, hour <dbl>, minute <dbl>
```

Il y a également la possibilité d'appliquer des sortes de filtres, ou des conditions à nos tableaux avec les tremes "starts_with", "ends_with", "contains" ou encore "matches" :

```
select(airports, starts_with("A"))
```

```
## # A tibble: 1,458 x 1
##      alt
##    <dbl>
##  1  1044
##  2   264
##  3   801
##  4   523
##  5    11
##  6  1593
##  7   730
##  8   492
##  9  1000
## 10   108
## # ... with 1,448 more rows
```

Dans cet exemple, j'ai affiché la seule colonne de ma table "airports" qui commençait par un "a".

Rename :

Le troisième verbe que nous allons voir est un dérivé de select et se nomme "rename".

Il nous permet de choisir certaines colonnes et de les renommer afin qu'elle soit plus lisible.

Par exemple :

```
rename(airports, altitude = alt, time_zone = tzone)
```

```
## # A tibble: 1,458 x 8
##    faa   name                   lat    lon altitude    tz dst   time_zone
##    <chr> <chr>                <dbl>  <dbl>    <dbl> <dbl> <chr> <chr>
##  1 04G   Lansdowne Airport     41.1  -80.6     1044    -5 A
America/New_Yo~
##  2 06A   Moton Field Municipa~  32.5  -85.7      264    -6 A
America/Chicago
##  3 06C   Schaumburg Regional    42.0  -88.1      801    -6 A
America/Chicago
##  4 06N   Randall Airport        41.4  -74.4      523    -5 A
America/New_Yo~
##  5 09J   Jekyll Island Airport  31.1  -81.4       11    -5 A
America/New_Yo~
##  6 0A9   Elizabethton Municip~  36.4  -82.2     1593    -5 A
America/New_Yo~
##  7 0G6   Williams County Airp~  41.5  -84.5      730    -5 A
America/New_Yo~
```

```
##  8 0G7   Finger Lakes Regiona~  42.9  -76.8      492    -5 A
America/New_Yo~
##  9 0P2   Shoestring Aviation ~  39.8  -76.6     1000    -5 U
America/New_Yo~
## 10 0S9   Jefferson County Intl  48.1 -123.       108    -8 A
America/Los_An~
## # ... with 1,448 more rows
```

Nous avons réussi à renommer 2 colonnes du tableau “airports” initiale.

Enfin, si les surnoms que nous souhaitons donner contiennent des espaces ou des caractères spéciaux tels que “é”, “è”, “ù”, etc, nous pouvons utiliser l’écriture ci-dessous :

```
rename(airports, "altitude du vol" = alt, "zone horaire" = tzone)
```

```
## # A tibble: 1,458 x 8
##    faa   name            lat    lon `altitude du vo~    tz dst   `zone
horaire`
##    <chr> <chr>         <dbl>  <dbl>            <dbl> <dbl> <chr> <chr>
##  1 04G   Lansdowne Ai~  41.1  -80.6             1044    -5 A
America/New_Yo~
##  2 06A   Moton Field ~  32.5  -85.7              264    -6 A
America/Chicago
##  3 06C   Schaumburg R~  42.0  -88.1              801    -6 A
America/Chicago
##  4 06N   Randall Airp~  41.4  -74.4              523    -5 A
America/New_Yo~
##  5 09J   Jekyll Islan~  31.1  -81.4               11    -5 A
America/New_Yo~
##  6 0A9   Elizabethton~  36.4  -82.2             1593    -5 A
America/New_Yo~
##  7 0G6   Williams Cou~  41.5  -84.5              730    -5 A
America/New_Yo~
##  8 0G7   Finger Lakes~  42.9  -76.8              492    -5 A
America/New_Yo~
##  9 0P2   Shoestring A~  39.8  -76.6             1000    -5 U
America/New_Yo~
## 10 0S9   Jefferson Co~  48.1 -123.              108    -8 A
America/Los_An~
## # ... with 1,448 more rows
```

Je tenais à remercier cette source pour sa grande aide ! Source

Vous pouvez retrouver tous mes dossiers juste ici !

Caption Mon Github

EVALUATION DU TRAVAIL EN QUESTION

Critère 1 : Visuel sur pdf 4/4 Tres Agreable a lire.

Critère 2 : Originalite du code 4/4 Jean a bien montre comment utiliser les differentes fonctions avec une demarche adapte

Critère 3 : Fonctionnalité du code 4/4 le code fonctionne.

Critère 4 : Lisibilité du code 4/4 Claire et lisible.

Critère 5 : Explications données 4/4 Toutes les explications sont claire sur chaque ligne de code.

CONCLUSION

Globalement un tres bon travail qui exlique tres bien dplyr. Un travail exemplaire avec une demarche pedagogique.