TD n° 5 : Tidyverse (partie 1)

Préambule. Dans un premier temps, on fait :

```
library(tidyverse) # qui contient tibble, dplyr et forcats, entre autre
# si problème, on peut les télécharger un à un

library(nycflights13) # pour les données
data(flights)
data(airports)
```

Exercice 1. Comparer les commandes suivantes :

```
round(exp(cos(pi ** 2)), digits = 2)

pi ** 2 %>% cos %>% exp %>% round(digits = 2)
```

Exercice 2. On considère le jeu de données `airports`.

1. Étudier l'aide R de `airports` pour comprendre les différents caractères considérés.
2. Demander un résumé descriptif, puis un résumé statistique des données.
3. Sélectionner la dixième ligne de `airports`.
4. Sélectionner l'aéroport avec l'altitude la plus basse.
5. Sélectionner les colonnes `name`, `lat` et `lon`.
6. Sélectionnez toutes les colonnes de `airports` sauf les colonnes `tz` et `tzone`.
7. Renommer la colonne `alt` en `altitude` et la colonne `tzone` en `fuseau_horaire`.
8. La colonne `alt` contient l'altitude de l'aéroport en pieds. Créer une nouvelle variable `alt_m` contenant l'altitude en mètres (on convertit des pieds en mètres en les divisant par 3.2808). Sélectionner dans la table obtenue uniquement les deux colonnes `alt` et `alt_m`.

Exercice 3. On considère le jeu de données `flights`.

1. À partir de `flights`, sélectionnez les vols du mois de juillet (on utilisera le caractère `month`).
2. Trier `flights` selon le retard au départ décroissant.
3. Sélectionner toutes les colonnes de `flights` dont les noms se terminent par `delay`.
4. Sélectionnez les vols avec un retard à l'arrivée compris entre 5 et 15 minutes (on considérera le caractère `arr_delay`).
5. Sélectionnez les vols des compagnies Delta, United et American, lesquelles sont codées par `DL`, `UA` et `AA` du caractère `carrier`.

6. En utilisant le pipe, sélectionner les vols à destination de San Francisco (modalité codée `SFO` du caractère `dest`) et trier-les selon le retard au départ décroissant (caractère `dep_delay`).
7. Sélectionnez les vols des mois de septembre et octobre, conservez les colonnes `dest` et `dep_delay`, créez un nouvelle colonne nommée `retard_h` contenant le retard au départ en heures, et conservez uniquement les 5 lignes avec les plus grandes valeurs de `retard_h`.
8. Affichez le nombre de vols par mois.
9. Trier `flights` selon le nombre de vols croissant.
10. Calculer la distance moyenne des vols selon l'aéroport de départ (caractère `origin`).
11. Calculer le nombre de vols à destination de Los Angeles (modalité codée `LAX`) pour chaque mois de l'année.
12. Calculer le nombre de vols selon le mois et la destination
13. Ne conserver, pour chaque mois, que la destination avec le nombre maximal de vols.
14. Calculer le nombre de vols selon le mois, et ajouter une colonne comportant le pourcentage de vols annuels réalisés par mois.
15. Calculer, pour chaque aéroport de départ et de destination, la durée moyenne des vols (caractère `air_time`). Pour chaque aéroport de départ, ne conserver que la destination avec la durée moyenne la plus longue.

Exercice 4. Reproduire et comprendre l'enjeu de commandes suivantes, utilisant des données de recensement des États Unis en 2012:

```
# Données: Recensement aux Etats-Unis en 2012
w = read.csv2("https://chesneau.users.lmno.cnrs.fr/Recensement_12.csv", row.names = 1)

head(w, 9)

str(w)

summary(w)

par(mfrow=c(2,2), mar = c(2, 2, 4, 0) + .1)
pie(table(w$SEXE), radius = 1, init.angle = 90,
    col = c("purple", "blue"), main = "Sexe")
hist(w$SAL_HOR,
     main = "Salaire horaire", col = "gray80",
     xlab = "", ylab = "")
p = 100*prop.table(table(w$NB_ENF))
barplot(p, main = "Nombre d'enfants au foyer")
p4 = 100*prop.table(table(w$REV_FOYER))
barplot(p4[c(1,3,4,2)], main = "Revenus du foyer")
```

```
round(cor(w[, c(1,5,9,10)], use = "complete.obs"), digits=2)

pairs(w[,c(1,5,9,10)])

# Activation du Tidyverse
# (library(tidyverse) est utilisée, normalement déjà téléchargée en préambule)

w %>% is.na %>% colMeans %>% mean %>% '*'(100)


wquanti = w %>% select_if(is.integer)

wquanti = w %>% select_if(is.double)

wquali = w %>% select_if(is.character)

ww = w %>% select(AGE, REGION)
# ou
ww = select(w, AGE, REGION)

ww %>% filter(AGE < 19)

w %>% mutate(X1 = 1:599, .before = AGE)

w %>% summarise(Mean_age = mean(AGE))

w %>%  arrange(AGE)

w %>% filter(AGE < 19) %>%
       filter( !is.na(SAL_HOR)) %>%
       summarise(Moyenne = mean(SAL_HOR), Ecart_type = sd(SAL_HOR))
```

Exercice 5.

1. Créer un vecteur noté `QI` composé de 20 valeurs générées par une loi normale de moyenne 100 et d'écart-type 15.
2. À partir de `QI`, en utilisant la fonction `ifelse`, créer un vecteur de caractères tel qui affiche `fort` si la valeur associée de `QI` est supérieure ou égale à 100, et `faible` sinon.
3. À partir de `QI`, en utilisant la fonction `case_when` du package `dplyr`, créer un vecteur de caractères noté `b` tel qui affiche `fort` si la valeur associée de `QI` est supérieure strictement à 115, `moyen` si la valeur associée de `QI` est supérieure strictement à 100 et inférieure ou égale à 115, et `faible` sinon.
4. Recoder les modalités de `b` tel que `faible` devienne `1`, `moyen` devienne `2` et `fort` devienne `3` en utilisant la fonction `fct_recode` du package `forcats`.