Test de management et d’analyse des données avec R

Jean-Luc BATABATI

2025-02-25

# Installation et importation des packages

Avant toute chose, nous veillerons à installer tous les packages qui servirons dans la suite puis les charger.

packages <- c("readr","haven","utils","dplyr", "kableExtra",   
 "gtsummary", "labelled", "survey", "cardx", "Hmisc")  
  
for (package in packages) {  
 if (!requireNamespace(package, quietly = TRUE)) {  
 install.packages(package)  
 }  
 library(package, character.only = TRUE)  
}

# Chargement des bases

Nous importons ici les deux bases baseline et endline puis la base ménage.

# Importation de la base de données baseline pour enfant  
C\_baseline <-haven::read\_dta("../data/food\_comp\_child\_baseline.dta")

# Importation d la base de données baseline pour mère  
M\_baseline <-haven::read\_dta("../data/food\_comp\_mother\_baseline.dta")

# Importatin de la base endline pour enfant  
C\_endline <-haven::read\_dta("../data/food\_comp\_child\_endline.dta")

# Importation de la base endline pour mère  
M\_endline <-haven::read\_dta("../data/food\_comp\_mother\_endline.dta")

#Importation de la base ménage  
Base\_men <- haven::read\_dta("../data/base\_menage.dta")

# Structure des bases

La base ménage contient 1065 observations et 21 variables. La base endline pour mère contient 4256 observations et 17 variables. La base base endline pour enfant contient 4256 observations et 17 variables. La base baseline pour mère contient 4256 observations et 17 variables. La base baseline pour enfant contient 4256 observations et 17 variables.

# Partie 1 : Gestion et nettoyage des bases de données

## 1- Vérifions la présence de doublons dans les bases de données Baseline, Endline et ménage et supprimons les doublons si nécessaire

### Vérification des doublons dans la base ménage

Dans cette base la clé d’identification est l’ID du ménage. Donc nous allons vérifier si un ID ne se repète pas

# On regroupe suivant l'ID du ménage et on filtre pour ceux qui sont supéieur à 1  
Base\_men %>%  
 group\_by(hhid) %>%  
 filter(n() > 1)

## # A tibble: 2 × 21  
## # Groups: hhid [1]  
## regionid communeid villageid hhid hhsize poly hh\_primary s1\_q2 s1\_q4a  
## <dbl> <dbl> <dbl> <chr> <dbl> <dbl+l> <dbl+lbl> <dbl+l> <dbl>  
## 1 1 1 15 4953484… 6 0 [Non] 0 [Non] 1 [Mal… 65  
## 2 1 1 15 4953484… 6 0 [Non] 0 [Non] 1 [Mal… 65  
## # ℹ 12 more variables: s2\_q1 <dbl+lbl>, s2\_q2 <dbl+lbl>, s2\_q4 <dbl+lbl>,  
## # s29\_q1 <dbl+lbl>, demgrp1 <dbl>, demgrp2 <dbl>, demgrp3 <dbl>,  
## # demgrp4 <dbl>, demgrp5 <dbl>, dependencyratio <dbl>, hfias\_score <dbl>,  
## # T1 <dbl+lbl>

# On peut tout simplement compter les occurences  
Base\_men %>%  
 count(hhid, sort = TRUE)

## # A tibble: 1,064 × 2  
## hhid n  
## <chr> <int>  
## 1 4953484948 2  
## 2 4948484848535052 1  
## 3 4948484848535053 1  
## 4 4948484848535055 1  
## 5 4948484848535056 1  
## 6 4948484848535151 1  
## 7 4948484848535152 1  
## 8 4948484848535154 1  
## 9 4948484848535155 1  
## 10 4948484848535253 1  
## # ℹ 1,054 more rows

On constate que le ménage avec l’ID 4953484948 se repète deux fois. Donc nous allons supprimer une observation

### Suppression des doublons dans la base ménage en gardant les premières

# Selection unique  
Base\_men <- Base\_men %>% distinct()

Dans les bases suivantes nous vérifierons les doublons sur toutes les colonnes

### Vérification des doublons dans la base baseline pour mère

# Somme des doublons  
sum(duplicated(M\_baseline))

## [1] 0

Dans la base baseline pour mère, il n’y a pas de doublons.

### Vérification des doublons dans la base baseline pour enfant

sum(duplicated(C\_baseline))

## [1] 0

Dans la base baseline pour enfant, il n’y a pas de doublons.

### Vérification des doublons dans la base endline pour mère

sum(duplicated(M\_endline))

## [1] 0

Dans la base endline pour mère, il n’y a pas de doublons.

### Vérification des doublons dans la base baseline pour enfant

sum(duplicated(C\_endline))

## [1] 0

Dans la base endline pour enfant, il n’y a pas de doublons.

## 2- Vérification de la cohérence des noms des variables. Apportons les corrections si nécessaire

### Nom des colonnes dans chaque base

print("Noms des colonnes de la Base baseline pour enfant")

## [1] "Noms des colonnes de la Base baseline pour enfant"

colnames(C\_baseline)

## [1] "regionid" "communeid" "villageid" "hhid" "round"   
## [6] "s1\_q0" "s1\_q1" "s1\_q2" "energ\_kcal" "protein\_g"   
## [11] "lipid\_tot\_g" "calcium\_mg" "iron\_mg" "zinc\_mg" "vit\_b6\_mg"   
## [16] "vit\_b12\_mcg" "vit\_c\_mg"

print("Noms des colonnes de la Base baseline pour mère")

## [1] "Noms des colonnes de la Base baseline pour mère"

colnames(M\_baseline)

## [1] "regionid" "communeid" "villageid" "hhid" "round"   
## [6] "s1\_q0" "s1\_q1" "s1\_q2" "V1" "protein\_g"   
## [11] "lipid\_tot\_g" "calcium\_mg" "iron\_mg" "V9" "vit\_b6\_mg"   
## [16] "vit\_b12\_mcg" "vit\_c\_mg"

print("Noms des colonnes de la Base endline pour enfant")

## [1] "Noms des colonnes de la Base endline pour enfant"

colnames(C\_endline)

## [1] "regionid" "communeid" "villageid" "hhid" "round"   
## [6] "s1\_q0" "s1\_q1" "s1\_q2" "energ\_kcal" "protein\_g"   
## [11] "lipid\_tot\_g" "calcium\_mg" "iron\_mg" "zinc\_mg" "vit\_b6\_mg"   
## [16] "vit\_b12\_mcg" "vit\_c\_mg"

print("Noms des colonnes de la Base endline pour mère")

## [1] "Noms des colonnes de la Base endline pour mère"

colnames(M\_endline)

## [1] "regionid" "communeid" "villageid" "hhid" "round"   
## [6] "s1\_q0" "s1\_q1" "s1\_q2" "energ\_kcal" "protein\_g"   
## [11] "lipid\_tot\_g" "calcium\_mg" "iron\_mg" "zinc\_mg" "vit\_b6\_mg"   
## [16] "vit\_b12\_mcg" "vit\_c\_mg"

### Vérifions si les noms sont les mêmes, même si l’ordre est différent

Je m’assure que les varaibles de la base baseline pour enfant sont les même avec les autres bases. Cette commande renvois true si c’est le cas.

#### Cohérence entre base baseline pour enfant et endline pour enfant

# Cohérence des variables entre base baseline pour enfant et endline pour enfant  
setequal(names(C\_baseline), names(C\_endline))

## [1] TRUE

On a true. Donc les variables sont les mêmes.

#### Cohérence entre base baseline pour enfant et baseline pour mère

setequal(names(C\_baseline), names(M\_baseline))

## [1] FALSE

On a false. Donc il y a une différence. Essayons de détecter la différence avant de renommer la variable.

# Vérifions la différence  
setdiff(names(C\_baseline), names(M\_baseline))

## [1] "energ\_kcal" "zinc\_mg"

setdiff(names(M\_baseline), names(C\_baseline))

## [1] "V1" "V9"

On constate que les variables “energ\_kcal” et “zinc\_mg” sont dans la base basline pour enfant mais absentes dans la base baseline pour mère. De même, les variables “V1” et “V9” sont dans la base baseline pour mère mais absentes dans la base basline pour enfant. Nous allons renommer celles de la base baseline pour mère pour assurer la cohérence.

# Renommer les variables différentes  
M\_baseline <- M\_baseline %>% rename(  
 energ\_kcal = V1,  
 zinc\_mg = V9  
)

#### Cohérence entre base baseline pour enfant et endline pour mère

setequal(names(C\_baseline), names(M\_endline))

## [1] TRUE

On a true. Donc les variables sont les mêmes.

## 3- Certaines variables peuvent contenir des données manquantes, cependant suivant une logique bien établie, il est possible de corriger ces données manquantes. Veuillez vérifier soigneusement les données et corriger les données manquantes de certaines variables si possibles.

### Valeurs manquantes pour la base ménages

# Compter les valeurs manquantes  
sum(is.na(Base\_men))

## [1] 1

colSums(is.na(Base\_men))

## regionid communeid villageid hhid hhsize   
## 0 1 0 0 0   
## poly hh\_primary s1\_q2 s1\_q4a s2\_q1   
## 0 0 0 0 0   
## s2\_q2 s2\_q4 s29\_q1 demgrp1 demgrp2   
## 0 0 0 0 0   
## demgrp3 demgrp4 demgrp5 dependencyratio hfias\_score   
## 0 0 0 0 0   
## T1   
## 0

On a une valeur manquante pour la base ménage. Il s’agit de la variable communeid. Nous avons pas les moyens pour corriger cette valeur.

### Valeurs manquantes pour la base baseline pour enfant

sum(is.na(C\_baseline))

## [1] 3474

colSums(is.na(C\_baseline))

## regionid communeid villageid hhid round s1\_q0   
## 0 0 0 0 0 0   
## s1\_q1 s1\_q2 energ\_kcal protein\_g lipid\_tot\_g calcium\_mg   
## 0 0 386 386 386 386   
## iron\_mg zinc\_mg vit\_b6\_mg vit\_b12\_mcg vit\_c\_mg   
## 386 386 386 386 386

Ici nous avons 3474 valeurs manquantes. On peut aussi voir la répartition de ces valeurs manquantes

### Valeurs manquantes pour la base baseline pour mère

sum(is.na(M\_baseline))

## [1] 6069

colSums(is.na(M\_baseline))

## regionid communeid villageid hhid round s1\_q0   
## 4 8 0 0 0 0   
## s1\_q1 s1\_q2 energ\_kcal protein\_g lipid\_tot\_g calcium\_mg   
## 0 0 673 673 673 673   
## iron\_mg zinc\_mg vit\_b6\_mg vit\_b12\_mcg vit\_c\_mg   
## 673 673 673 673 673

Ici nous avons 6090 valeurs manquantes. On peut aussi voir la répartition de ces valeurs manquantes

### Valeurs manquantes pour la base endline pour enfant

sum(is.na(C\_endline))

## [1] 4788

colSums(is.na(C\_endline))

## regionid communeid villageid hhid round s1\_q0   
## 0 0 0 0 0 0   
## s1\_q1 s1\_q2 energ\_kcal protein\_g lipid\_tot\_g calcium\_mg   
## 0 0 532 532 532 532   
## iron\_mg zinc\_mg vit\_b6\_mg vit\_b12\_mcg vit\_c\_mg   
## 532 532 532 532 532

Ici nous avons 4788 valeurs manquantes. On peut aussi voir la répartition de ces valeurs manquantes

### Valeurs manquantes pour la base endline pour mère

sum(is.na(M\_endline))

## [1] 5976

colSums(is.na(M\_endline))

## regionid communeid villageid hhid round s1\_q0   
## 0 0 0 0 0 0   
## s1\_q1 s1\_q2 energ\_kcal protein\_g lipid\_tot\_g calcium\_mg   
## 0 0 664 664 664 664   
## iron\_mg zinc\_mg vit\_b6\_mg vit\_b12\_mcg vit\_c\_mg   
## 664 664 664 664 664

Ici nous avons 5976 valeurs manquantes. On peut aussi voir la répartition de ces valeurs manquantes.

## 4. Consommation d’énergie moyenne à chaque repas pour l’ensemble des mères lors de l’enquête de base

M\_baseline %>% summarise(Energie\_moy = mean(energ\_kcal, na.rm = TRUE))

## # A tibble: 1 × 1  
## Energie\_moy  
## <dbl>  
## 1 715.

La consommation d’énergie moyenne à chaque repas pour l’ensemble des mères est de 714,7165

## 5- Sauvegarde des données sous un autre nom

write\_dta(C\_endline, "../data/child\_endline\_v1.dta")  
write\_dta(M\_endline, "../data/mother\_endline\_v1.dta")  
write\_dta(C\_baseline, "../data/child\_baseline\_v1.dta")  
write\_dta(M\_baseline, "../data/mother\_baseline\_v1.dta")  
write\_dta(Base\_men, "../data/base\_menage\_final.dta")

### Importation des nouvelles base

child\_endline\_v1 <- haven::read\_dta("../data/child\_endline\_v1.dta")  
mother\_endline\_v1 <- haven::read\_dta("../data/mother\_endline\_v1.dta")  
child\_baseline\_v1 <- haven::read\_dta("../data/child\_baseline\_v1.dta")  
mother\_baseline\_v1 <- haven::read\_dta("../data/mother\_baseline\_v1.dta")  
base\_menage\_final <- haven::read\_dta("../data/base\_menage\_final.dta")

# Partie 2 : Empilement et Fusion des données

### 1- i) Empilons les bases de données des mères (mother\_baseline\_v1.dta) et des enfants (child\_baseline\_v1.dta)

# Ici nous allons faire un appends étant donné que les variables sont les même  
baseline <- rbind(child\_baseline\_v1, mother\_baseline\_v1)

### 1- ii) Renommons toutes les variables de consommation energ\_kcal jusqu’à vit\_c\_mcg en ajoutant le suffixe \_b pour faire référence à l’enquête Baseline

baseline <- baseline %>%  
 rename\_with(~ paste0(., "\_b"),   
 .cols = energ\_kcal:vit\_c\_mg)

### 1- iii) Créeons une base de données qui résume les consommations journalières totales par individu (somme des 4 repas) pour l’énergie et tous les nutriments en utilisant la commande merge.

# Sélectionner uniquement les variables à sommer  
variables <- c("energ\_kcal\_b", "protein\_g\_b", "lipid\_tot\_g\_b",   
 "calcium\_mg\_b", "iron\_mg\_b", "zinc\_mg\_b")  
  
# Vérifier si toutes les variables existent dans la base  
variables <- intersect(variables, names(baseline))  
  
# Agréger les données par hhid en sommant les valeurs  
baseline2 <- baseline %>%  
 group\_by(hhid, s1\_q2) %>%  
 summarise(across(all\_of(variables), sum, na.rm = TRUE))

### 1-iv) Sauvegarde la base de données finale contenant seulement les hhid s1\_q2 energ\_kcal protein\_g lipid\_tot\_g calcium\_mg iron\_mg zinc\_mg sous le nom « baseline\_final.dta ».

# Sélectionner uniquement les variables nécessaires  
baseline2 <- baseline2 %>%  
 select(hhid, s1\_q2, energ\_kcal\_b, protein\_g\_b, lipid\_tot\_g\_b, calcium\_mg\_b, iron\_mg\_b, zinc\_mg\_b)  
  
# Sauvegarder la base de données sous le nom 'baseline\_final.dta'  
write\_dta(baseline, "../data/baseline\_final.dta")

## 2) Endline

### 2-i) Empilons les bases de données des mères (mother\_endline\_v1.dta) et des enfants (child\_endline\_v1.dta)

# Ici nous allons faire un appends étant donné que les variables sont les même  
endline <- rbind(child\_endline\_v1, mother\_endline\_v1)

#### Renommons toutes les variables de consommation energ\_kcal jusqu’à vit\_c\_mcg en ajoutant le suffixe \_e pour faire référence à l’enquête Endline.

endline <- endline %>%  
 rename\_with(~ paste0(., "\_e"),   
 .cols = energ\_kcal:vit\_c\_mg)

### ii) Créeons une base de données qui résume les consommations journalières totales par individu (somme des 4 repas) pour l’énergie et tous les nutriments en utilisant la commande collapse

# Sélectionner uniquement les variables à sommer  
variables <- c("energ\_kcal\_e", "protein\_g\_e", "lipid\_tot\_g\_e",   
 "calcium\_mg\_e", "iron\_mg\_e", "zinc\_mg\_e")  
  
# Vérifier si toutes les variables existent dans la base  
variables <- intersect(variables, names(endline))  
  
# Agréger les données par hhid en sommant les valeurs  
endline2 <- endline %>%  
 group\_by(hhid, s1\_q2) %>%  
 summarise(across(all\_of(variables), sum, na.rm = TRUE))

### iii) Sauvegardons la base de données finale contenant seulement les hhid s1\_q2 energ\_kcal protein\_g lipid\_tot\_g calcium\_mg iron\_mg zinc\_mg sous le nom « endline\_final.dta ».

# Sélectionner uniquement les variables nécessaires  
endline2 <- endline2 %>%  
 select(hhid, s1\_q2, energ\_kcal\_e, protein\_g\_e, lipid\_tot\_g\_e, calcium\_mg\_e, iron\_mg\_e, zinc\_mg\_e)  
  
# Sauvegarder la base de données sous le nom 'baseline\_final.dta'  
write\_dta(endline2, "../data/endline\_final.dta")

## 3) Fusionnez les données baseline\_final.dta et endline\_final.dta. La base de données résultant de cette fusion doit comporter le même nombre d’observations que les bases baseline\_final.dta et endline\_final.dta.

## Importation des deux base

endline\_final <- haven::read\_dta("../data/endline\_final.dta")  
baseline\_final <- haven::read\_dta("../data/baseline\_final.dta")

## Fusion des deux bases

Base <- baseline\_final %>%  
 dplyr::left\_join(endline\_final, by = c("hhid", "s1\_q2"))

### 4) En utilisant la base base\_menage\_final, fusionnez les informations sur les caractéristiques des ménages

Base\_finale <- baseline\_final %>%  
 dplyr::left\_join(endline\_final, by = c("hhid", "s1\_q2"))

# Partie 3 : Statistiques descriptives avec le package gtsummary