LATERITE DATA QUALITY ANALYST

LANGAT ERICK

2023-12-30

DATA QUALITY ANALYST ASSESSEMENT

Question:1

1. Using the variables names and values labels provided in Table 1 below, please label:

a. all variables in the main and roster datasets. (10 marks)

b. the values of all categorical variables in both the main and roster datasets. (5marks)

Main Dataset:

Variable Labels:

- province_id: Province code
- province_name: Province of residence
- district_id: District code
- district_name: District of residence
- sector_id: Sector code
- sector_name: Sector of residence
- cell_id: Cell code
- cell_name: Cell of residence
- village_id: Village code
- village_name: Village of residence
- list_avail: Ubudehe list availability
- list_source: Source of the ubudehe list
- ubudehe_contact: Ubudehe list contact person
- contact_role: Role of the ubudehe list contact person
- contact_phone: Phone number of the ubudehe list contact person
- num_households: Number of households in ubudehe category 1

- hh_head_position: Position of household head on ubudehe list
- name_hh_head: Name of the household head
- nid_hh_head: National ID number of the household head
- phone_hh_head: Phone number of the household head
- gps: GPS coordinates of where the list was obtained
- parent_key: Unique ID
- **Categorical variables**

  list_avail:

- 0: No
- 1: Yes

  list_source:

- 1: Sector Executive Officer
- 2: Cell Executive Officer
- 3: Village Leader

  contact_role:

- 1: Village Leader
- 2: Village in-charge of security
- 3: Village Community Health Worker

Roster Dataset:

Variable Labels:

- hh_head_position: Position of household head on ubudehe list
- name_hh_head: Name of the household head
- nid_hh_head: National ID number of the household head
- phone_hh_head: Phone number of the household head

Categorical Variable Values:

There are no specific categorical variables in the roster dataset based on the information provided.

Question:2

2. Using the values and value labels in Table 1 (Variable description) below, create a single variable for each of the location variables (province, district, sector, cell, and village) that displays the value label but also has the value embedded. (15 marks)

```
#Load Packages
suppressPackageStartupMessages(require(tidyverse))

#Load data
#main dataset
main <- read.csv("C:/Users/langa/OneDrive/Desktop/DATA CLEANING R
PROGRAMMING/main_dataset.csv")
colnames(main)#column names

##  [1] "province_id"     "province_name"   "district_id"     "district_name"
##  [5] "sector_id"       "sector_name"     "cell_id"         "cell_name"
##  [9] "village_id"      "village_name"    "list_available"  "list_source"
## [13] "ubudehe_contact" "contact_role"    "contact_phone"
"num_households"
## [17] "gpslatitude"     "gpslongitude"    "gpsaltitude"     "gpsaccuracy"
## [21] "parent_key"

#single variable for each of the location variables (province, district,
sector, cell, and
# village) that displays the value label but also has the value embedded.

# For Province
main$province_name <- paste(main$province_name, "(", main$province_id, ")",
sep = " ")

# For District
main$district_name <- paste(main$district_name, "(", main$district_id, ")",
sep = " ")

# For Sector
main$sector_name <- paste(main$sector_name, "(", main$sector_id, ")", sep = "
")

# For Cell (replace 'cell' with the actual column name)
main$cell_name <- paste(main$cell_name, "(", main$cell_id, ")", sep = " ")

# For Village (replace 'village' with the actual column name)
main$village_name <- paste(main$village_name, "(", main$village_id, ")", sep
= " ")

# Display the updated data frame
# print(main)
# view(main)
```

```r
#Roster dataset
roster <- read.csv("C:/Users/langa/OneDrive/Desktop/DATA CLEANING R
PROGRAMMING/roster_dataset.csv")# importing dataset
colnames(roster)# These are the column names
```

```
## [1] "hh_head_position" "name_hh_head"     "nid_hh_head"
"phone_hh_head"
## [5] "parent_key"
```

Question:3

3. Please create a dataset combining the main and roster dataset so that we have one observation per village with all household heads as well as their personal information contained in the roster. Save this dataset as "merged_yourname_yyyymmdd".

```r
# Merge the main and roster datasets based on the common column "parent_key"
merged_data <- inner_join(main, roster, by = "parent_key")# joining the 2-
data using a common ID

# The inner_join() function from the 'dplyr' package is used to merge the
main and roster datasets based on the common column 'parent_key'. This
function performs an inner join, which means it keeps only the rows that have
matching values in both

# Save the merged dataset as a CSV file
write.csv(merged_data, file = "merged_ericklangat_20231229.csv",
                row.names = FALSE)#Save the results in a CSV._ Format

# Display the merged dataset
 head(merged_data,4)# check the First 4 Rows of the data
```

```
##   province_id province_name district_id     district_name sector_id
## 1           1  Kigali ( 1 )          11 Nyarugenge ( 11 )      1101
## 2           1  Kigali ( 1 )          11 Nyarugenge ( 11 )      1101
## 3           1  Kigali ( 1 )          11 Nyarugenge ( 11 )      1101
## 4           1  Kigali ( 1 )          11 Nyarugenge ( 11 )      1101
##       sector_name cell_id           cell_name village_id
village_name
## 1 Gitega ( 1101 )  110101 Akabahizi ( 110101 )   11010102 Gihanga (
11010102 )
## 2 Gitega ( 1101 )  110101 Akabahizi ( 110101 )   11010102 Gihanga (
11010102 )
## 3 Gitega ( 1101 )  110101 Akabahizi ( 110101 )   11010102 Gihanga (
11010102 )
## 4 Gitega ( 1101 )  110101 Akabahizi ( 110101 )   11010102 Gihanga (
11010102 )
##   list_available list_source ubudehe_contact contact_role contact_phone
## 1              1           3         ndakaza            2     788004184
## 2              1           3         ndakaza            2     788004184
## 3              1           3         ndakaza            2     788004184
```

```
## 4              1           3         ndakaza            2     788004184
##   num_households gpslatitude gpslongitude gpsaltitude gpsaccuracy
## 1              5   -1.943086     30.05912           0         200
## 2              5   -1.943086     30.05912           0         200
## 3              5   -1.943086     30.05912           0         200
## 4              5   -1.943086     30.05912           0         200
##                                  parent_key hh_head_position name_hh_head
## 1 uuid:7ae0a1ab-0f9f-4708-b787-5f9a9209ae23                1     mwitende
## 2 uuid:7ae0a1ab-0f9f-4708-b787-5f9a9209ae23                2   ruhumuriza
## 3 uuid:7ae0a1ab-0f9f-4708-b787-5f9a9209ae23                3      kabanda
## 4 uuid:7ae0a1ab-0f9f-4708-b787-5f9a9209ae23                4      kangabe
##    nid_hh_head phone_hh_head
## 1 1.197194e+15     797352301
## 2 1.194808e+15     729858153
## 3 1.196502e+15     726559120
## 4 1.200033e+15     723251934
```

```
names(merged_data)# Checking the names of the Merged_dataset(Main & Roster)
```

```
##  [1] "province_id"      "province_name"    "district_id"
"district_name"
##  [5] "sector_id"        "sector_name"      "cell_id"          "cell_name"
##  [9] "village_id"       "village_name"     "list_available"
"list_source"
## [13] "ubudehe_contact"  "contact_role"     "contact_phone"
"num_households"
## [17] "gpslatitude"      "gpslongitude"     "gpsaltitude"
"gpsaccuracy"
## [21] "parent_key"       "hh_head_position" "name_hh_head"
"nid_hh_head"
## [25] "phone_hh_head"
```

```
#Rename Two Variables "list_available" to "list_avail"& 'gpsaccuracy' to
'gps'
merged_data$list_avail <- merged_data$list_available #Renaming the column
merged_data$gps <- merged_data$gpsaccuracy# Renaming the column
```

Question:4

4. Organize the data set.

a. Order the variables as per Table 1 (Variable description) below. (5 marks)

b. Make sure all variables are labeled properly as per Table 1 (Variable description) below. Save this dataset as “clean_yourname_yyyymmdd”. (10 marks)

```
# Reorder the variables as per description provided
ordered_variables <- c( "province_id", "province_name", "district_id",
"district_name", "sector_id", "sector_name",
  "cell_id", "cell_name", "village_id", "village_name", "list_avail",
"list_source",
```

```
  "ubudehe_contact", "contact_role", "contact_phone", "num_households",
  "hh_head_position", "name_hh_head", "nid_hh_head", "phone_hh_head", "gps",
"parent_key")

ordered_data <- merged_data %>%  select(ordered_variables)# Selecting the
required columns only and ordering them as per the description

## Warning: Using an external vector in selections was deprecated in
tidyselect 1.1.0.
## ℹ Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(ordered_variables)
##
##   # Now:
##   data %>% select(all_of(ordered_variables))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

# Save the ordered  dataset to #Clean data as a CSV file
write.csv(ordered_data, file = "clean_ericklangat_20231230.csv", row.names =
FALSE)#Save the results in a csv-format
```

Question:5

5. To aid the field team in drafting a data collection field plan, please extract lists of households per district in .xlsx format, with each sector in each of the 3 districts on a separate worksheet. Save your do-file as "yourname_yyymmdd". (25 marks)

```
dim(ordered_data)#check number of rows and columns before Removing NA'S

## [1] 480  22

order_d= ordered_data
order_d <- na.omit(order_d) %>%  distinct()#Remove Missing Vlues and the
duplicates
dim(order_d)# Check the dimension of the new data after Removing NA'S

## [1] 471  22

#We Group by district and sector
Gasabo_data <- order_d %>%
      select(district_name, sector_name, num_households) %>%
              filter(district_name=="Gasabo") %>%
                      group_by(district_name, sector_name)

# Summarize the data as per the description provided
summary_data1 <- Gasabo_data %>%
                    summarise(num_households = sum(num_households))
```

```
## `summarise()` has grouped output by 'district_name'. You can override
using the
## `.groups` argument.

summary_data1 %>%  head()

## # A tibble: 0 × 3
## # Groups:   district_name [0]
## # ℹ 3 variables: district_name <chr>, sector_name <chr>, num_households
<int>

#Group by district and sector
Nyarugenge_data <- order_d %>% select(district_name, sector_name,
num_households) %>%
             filter(district_name=="Nyarugenge") %>%
                   group_by(district_name, sector_name)

# Summarize the data as needed
summary_data2 <- Nyarugenge_data %>%
                          summarise(num_households = sum(num_households))

## `summarise()` has grouped output by 'district_name'. You can override
using the
## `.groups` argument.

summary_data2 %>%  head()

## # A tibble: 0 × 3
## # Groups:   district_name [0]
## # ℹ 3 variables: district_name <chr>, sector_name <chr>, num_households
<int>

#We Group by district and sector
Kicukiro_data <- order_d %>% select(district_name, sector_name,
num_households) %>%
  filter(district_name=="Kicukiro") %>%
  group_by(district_name, sector_name)

# Summarize the data as needed
summary_data3 <- Kicukiro_data %>%
          summarise(num_households = sum(num_households))

## `summarise()` has grouped output by 'district_name'. You can override
using the
## `.groups` argument.

summary_data3 %>%  head()

## # A tibble: 0 × 3
## # Groups:   district_name [0]
```

```
## # ⓘ 3 variables: district_name <chr>, sector_name <chr>, num_households
<int>

# Save the  dataset to # a CSV file the convert to xlsx format
write.csv(summary_data1, file = "ericklangat0_20231230.csv", row.names =
FALSE)
write.csv(summary_data2, file = "ericklangat1_20231230.csv", row.names =
FALSE)
write.csv(summary_data3, file = "ericklangat2_20231230.csv", row.names =
FALSE)
```