```
# Title ------------------------------------------------------------------------------

# SOCIO-ECONOMIC DATA QUALITY ASSURANCE AND QUALITY CHECK SCRIPT


# Contributor ------------------------------------------------------------------------

# Md. Akhter Hossain, FAO of the UN/Institute of Forestry and Environmental Sciences Chittagong University


# checking working directory

getwd()


#remove plots

dev.off()


#erage memory

rm(list=ls())


#Define working directory

setwd("C:/Users/User/Google Drive/3_Other works/5_NFI_FAO/Socio-economic data")

#setwd("C:/Users/Md. Akhter Hossain/Google Drive/3_Other works/5_NFI_FAO/Socio-economic data")


### Import data tables----------------------------------------------------------------

# Field data

age <- read.csv("./Socio-DB/age_class_gender.csv", header = T, stringsAsFactors = F)

conflict <- read.csv("./Socio-DB/conflict_tree_forest.csv", header = T, stringsAsFactors = F)

cyclone <- read.csv("./Socio-DB/damage_cyclone.csv", header = T, stringsAsFactors = F)

entry_fee <- read.csv("./Socio-DB/entry_fee_product_srf.csv", header = T, stringsAsFactors = F)

gen_pdt <- read.csv("./Socio-DB/gen_age_pdt.csv", header = T, stringsAsFactors = F)

gen_sell <- read.csv("./Socio-DB/gen_age_sell.csv", header = T, stringsAsFactors = F)

dependence <- read.csv("./Socio-DB/hh_dependence.csv", header = T, stringsAsFactors = F)
```

```
distrb <- read.csv("./Socio-DB/hh_disturbance.csv", header = T, stringsAsFactors = F)
gen_age <- read.csv("./Socio-DB/hh_gender_age.csv", header = T, stringsAsFactors = F)
tree_emp <- read.csv("./Socio-DB/hh_tree_emp.csv", header = T, stringsAsFactors = F)
hh_info <- read.csv("./Socio-DB/household.csv", header = T, stringsAsFactors = F)
hum_distrb <- read.csv("./Socio-DB/human_made_disturbances.csv", header = T, stringsAsFactors = F)
land <- read.csv("./Socio-DB/land_type_size.csv", header = T, stringsAsFactors = F)
nat_distrb <- read.csv("./Socio-DB/natural_disturbance.csv", header = T, stringsAsFactors = F)
for_prodt <- read.csv("./Socio-DB/primary_tree_forest_products.csv", header = T, stringsAsFactors = F)
pros_pordt <- read.csv("./Socio-DB/processed_tree_forest_product.csv", header = T, stringsAsFactors =
F)
wat_collct <- read.csv("./Socio-DB/water_collection.csv", header = T, stringsAsFactors = F)
wat_treat <- read.csv("./Socio-DB/water_treatment.csv", header = T, stringsAsFactors = F)
# Auxiliary data
lc_area <- read.csv("./Socio-DB/land_cover_class_area.csv", header = T, stringsAsFactors = F)
zone_hh <- read.csv("./Socio-DB/zone_hh_population_number.csv", header = T, stringsAsFactors = F)
locations <- read.csv("./Socio-DB/sampling_points_SE.csv",header = T, stringsAsFactors = F)


#read data 2nd phase tables
age <- read.csv("./2nd phase/age_class_gender.csv", header = T, stringsAsFactors = F)
conflict <- read.csv("./2nd phase/conflict_tree_forest.csv", header = T, stringsAsFactors = F)
cyclone <- read.csv("./2nd phase/damage_cyclone.csv", header = T, stringsAsFactors = F)
entry_fee <- read.csv("./2nd phase/entry_fee_product_srf.csv", header = T, stringsAsFactors = F)
gen_pdt <- read.csv("./2nd phase/gen_age_pdt.csv", header = T, stringsAsFactors = F)
gen_sell <- read.csv("./2nd phase/gen_age_sell.csv", header = T, stringsAsFactors = F)
dependence <- read.csv("./2nd phase/hh_dependence.csv", header = T, stringsAsFactors = F)
distrb <- read.csv("./2nd phase/hh_disturbance.csv", header = T, stringsAsFactors = F)
gen_age <- read.csv("./2nd phase/hh_gender_age.csv", header = T, stringsAsFactors = F)
tree_emp <- read.csv("./2nd phase/hh_tree_emp.csv", header = T, stringsAsFactors = F)
```

```
hh_info <- read.csv("./2nd phase/household.csv", header = T, stringsAsFactors = F)

hum_distrb <- read.csv("./2nd phase/human_made_disturbances.csv", header = T, stringsAsFactors = F)

land <- read.csv("./2nd phase/land_type_size.csv", header = T, stringsAsFactors = F)

nat_distrb <- read.csv("./2nd phase/natural_disturbance.csv", header = T, stringsAsFactors = F)

for_prodt <- read.csv("./2nd phase/primary_tree_forest_products.csv", header = T, stringsAsFactors = F)

pros_pordt <- read.csv("./2nd phase/processed_tree_forest_product.csv", header = T, stringsAsFactors
= F)

wat_collct <- read.csv("./2nd phase/water_collection.csv", header = T, stringsAsFactors = F)

wat_treat <- read.csv("./2nd phase/water_treatment.csv", header = T, stringsAsFactors = F)

othr_income <- read.csv("./2nd phase/other_income_sources.csv", header=T, stringsAsFactors = F)

lc_area <- read.csv("./Socio-DB/land_cover_class_area.csv", header = T, stringsAsFactors = F)

zone_hh <- read.csv("./Socio-DB/zone_hh_population_number.csv", header = T, stringsAsFactors = F)

locations <- read.csv("./Socio-DB/sampling_points_SE.csv",header = T, stringsAsFactors = F)


#combined data

age <- read.csv("./combined_03-10-18/age_class_gender_2nd.csv", header = T, stringsAsFactors = F)

conflict <- read.csv("./combined_03-10-18/conflict_tree_forest_2nd.csv", header = T, stringsAsFactors =
F)

cyclone <- read.csv("./combined_03-10-18/damage_cyclone_2nd.csv", header = T, stringsAsFactors = F)

entry_fee <- read.csv("./combined_03-10-18/entry_fee_product_srf_2nd.csv", header = T,
stringsAsFactors = F)

gen_pdt <- read.csv("./combined_03-10-18/gen_age_pdt_2nd.csv", header = T, stringsAsFactors = F)

gen_sell <- read.csv("./combined_03-10-18/gen_age_sell_2nd.csv", header = T, stringsAsFactors = F)

dependence <- read.csv("./combined_03-10-18/hh_dependence_2nd_26_09_18.csv", header = T,
stringsAsFactors = F)

distrb <- read.csv("./combined_03-10-18/hh_disturbance_2nd.csv", header = T, stringsAsFactors = F)

gen_age <- read.csv("./combined_03-10-18/hh_gender_age_2nd.csv", header = T, stringsAsFactors = F)

tree_emp <- read.csv("./combined_03-10-18/hh_tree_emp_2nd.csv", header = T, stringsAsFactors = F)

hh_info <- read.csv("./combined_03-10-18/household_combined_2nd_checked.csv", header = T,
stringsAsFactors = F)
```

```
hum_distrb <- read.csv("./combined_03-10-18/human_made_disturbances_2nd.csv", header = T, stringsAsFactors = F)

land <- read.csv("./combined_03-10-18/land_type_size_2nd.csv", header = T, stringsAsFactors = F)

nat_distrb <- read.csv("./combined_03-10-18/natural_disturbance_2nd.csv", header = T, stringsAsFactors = F)

othr_income <- read.csv("./combined_03-10-18/other_income_sources_2nd.csv", header = T, stringsAsFactors = F)

for_prodt <- read.csv("./combined_03-10-18/primary_tree_forest_products_2nd_26_9_18.csv", header = T, stringsAsFactors = F)

pros_pordt <- read.csv("./combined_03-10-18/processed_tree_forest_product_2nd.csv", header = T, stringsAsFactors = F)

wat_collct <- read.csv("./combined_03-10-18/water_collection_2nd.csv", header = T, stringsAsFactors = F)

wat_treat <- read.csv("./combined_03-10-18/water_treatment_2nd.csv", header = T, stringsAsFactors = F)



#QC output data frame

output <- data.frame(step="Step",cat="Category",hh_unique_no=99999,issue="Issue", location_name_interviewer = 99,  stringsAsFactors = F)


#--------------------------- CHECK LIST ----------------------------------#

# 1 Age

# 1.1 checking double record of one age class for one individual hh

# 1.2 checking total number of family member

# 1.3 abnormal number of hh members

# 1.4 Respondents age invalid (>80 yrs or <18 yrs)


# 2 Primary product collection

# 2.1 Inconsistency regarding primary forest product collection

# 2.2 data missing: forest product sold

# 2.3 data missing: LCC (location_tree_forests)) data missing
```

2.4 Product sold unit price missing

2.5 Primary product sold price is abnormal

2.6 Product quantity distribution

2.7 inconsistency in primary product collected Vs sold

2.8 validity of primary product collection location

2.9 Collection of product from invalid LCC

3 Conflict

3.1 Conflict mentioned but conflict frequency and related other info is missed

4. Cyclone

4.1 No damage but economic loss mentioned

4.2 hh_damage zero but economic loss is NA instead of zero

4.3 hh_damage mentioned but economic loss is absent

4.4 Double entry of same hh_damage for single hh

4.5 Cyclone damage amount (BDT) > 500 000 TK per damage type per hh

4.6 Total damage amount (BDT) per hh is exceptionally high

5 Potable water collection

5.1 data missing in water collection table

5.1.1 Water collection location missing

5.1.2 hh_water_collect information missing

5.1.3 months_water data missing

5.1.4 hours_water data missing

5.1.5 amount_water data missing

5.1.6 cost_water_trans data missing

5.2 water collection months inconsistent (more than 12 or less than 6)

5.3 Water collection amount less than 5 litre per day

5.4 Water collection time (hrs/day) is exceptionally high

5.5 Water transportation cost unusual (1-30 tk/month)

6 Water treatment

6.1 cost_water_treat cost data is NA

6.2 Water is treated but treatment cost is 0

6.3 Water treatment cost unrealistic

Input: water_treatment.csv

Output: List of hh for which water treatment cost >/= 10 TK/litre

7 forest product entry fee

7.1 missing data in pass_permit_product, number_collection and

amount_entry (amount of pass permit per trip for each product)

input: entry_fee_product_srf.csv

output: list of hh where data is missing

7.2 Tree and forest product for which pass permit is required is inconsistent

Input: entry_fee_product_srf.csv; household.csv and

primary_tree_forest_products.csv

Output: The list of hh for which the pass_permit_product will not match

with the list of primary product collection

8 processed_tree_forest_product.csv

8.1 processed product produced but quanity not mentioned

Question: what is the unit of quanity?

8.2 Number of hh members involved in processing product is missing

and inconsistent

8.3 data missing in processed_tree_forest_product

8.3.1 months_processed data missing

8.3.2 days_processed data missing

8.3.3 hours_processed data missing

```
# 8.3.4 raw material buy data missing
# 8.3.5 raw material cost data missing
# 8.3.6 labor hire data missing
# 8.3.7 labor cost data missing
# 8.3.8 processed product selling data missing
# 8.4 Labor cost inconsistent
# 8.5 Processed product no sold but quantity sold, hh_processed_selling,
#     months_selling, days_selling, hours_selling, income_processed
#     cost_selling_processed, transport_other_cost data mentioned
# 8.6 Production cost is higher than the selling const of processed product
# 8.7 Raw material bought but buying cost not mentioned

# 9 Cyclone
# 9.1 HH damage under 96 category due to Cyclone
#     Input: cyclone (damage_cyclone.csv)
#     Output: list of hh_id with damage names under 96 category that need revision

#---------------------- END OF CHECK LIST ---------------------------------#

#Check 1: NAs or missing data
#Check 1.1: NAs in interview date
length(hh_info[which(is.na(hh_info$date_interview_day)),]$hh_unique_no)
#Check 1.2: NAs in location of interviewed household
length(hh_info[which(is.na(hh_info$location_household_location)),]$household_hh_unique_no)
length(hh_info[which(is.na(hh_info$location_local_admin_mauza_name)),]$household_hh_unique_no)
length(hh_info[which(is.na(hh_info$location_local_admin_village_name)),]$household_hh_unique_no)
length(hh_info[which(is.na(hh_info$location_zone)),]$household_hh_unique_no)
```

```
#-----------------------------------------------------------------------------#
# 1.1 checking double record of one age class for one individual hh
# Input: Age
# Output: list of hh id where same age class is double counted/input
#-----------------------------------------------------------------------------#
age$count <- 1
age_class_test <- aggregate(count~age_class+household_hh_unique_no, data=age, FUN=sum)



# select the problematic rows and put those on output
if (dim(age_class_test[which(age_class_test$count>1),])[1] == 0){
  output1.1 <- rbind(output, data.frame(step="Step 1.1",cat="Double record of age class of one hh",
                             hh_unique_no=99999,
                             issue="No problems found",
                             location_name_interviewer = "not applicable",
                             stringsAsFactors = F))
} else {
  age_class_test <-  age_class_test[which(age_class_test$count>1),]
  age_class_test$issue= "age class label for some hh double entered"
  age_class_test$step="Step 1.1"
  age_class_test$cat= "Double record of age class of one hh"
  age_class_test <- merge(age_class_test, hh_info[,c("hh_unique_no",
                                        "location_name_interviewer")],
                    by.x="household_hh_unique_no",
                    by.y = "hh_unique_no", all.x=T)
  names(age_class_test)[1] <- "hh_unique_no"
  output1.1 <- rbind(output, age_class_test[,c("step", "cat", "hh_unique_no",
                                 "issue", "location_name_interviewer")])
}
```

```
#-------------------------------------------------------------------------------#
#3.2 checking total number of family member
#    Input: Age
#    Output: List of plots
#-------------------------------------------------------------------------------#
#replace the NAs with zero
age[,4:6] <-
  apply(age[,4:6], 2, function(x){replace(x, is.na(x), 0)}) #here 2 indicates colums



#sum male, female and third gender to get total family member
age_summary <- aggregate(cbind(male, female, third_gender)~household_hh_unique_no, data=age,
                         FUN=sum)

age_summary$tot_fam_mem <- age_summary$male + age_summary$female +
  age_summary$third_gender



#top 10 families with highest members in decending order
hi_fam_mem <- age_summary[order(-age_summary$tot_fam_mem),] [1:10,]
low_fam_mem <- age_summary[order(age_summary$tot_fam_mem),] [1:10,]
third_gen_fam <- age_summary[order(-age_summary$third_gender),][1:2,]

#plot number of total family member Vs number of male from each family
#load necessary library
library(ggplot2)
```

```
ggplot(age_summary, aes(x=male, y=tot_fam_mem))+
  geom_point()+
  xlab("Number of male")+ylab("Total number of family member")


#add total family number from hh_info
age_summary <- merge(age_summary, hh_info[,c("hh_unique_no","demographic_info_hh_members")],
          by.x="household_hh_unique_no",
          by.y="hh_unique_no",
          all.x=T)


# select the problematic rows and put those on output
if (dim(age_summary[which(age_summary$tot_fam_mem!=
                    age_summary$demographic_info_hh_members),])[1] == 0){
  output1.2 <- rbind(output1.1, data.frame(step="Step 1.2",cat="Mismatch in family members' number
",
                             hh_unique_no=99999,
                             issue= "No problem found",
                             location_name_interviewer = "not applicable",
                             stringsAsFactors = F))
} else {
  fam_mem_mismatch <-  age_summary[which(age_summary$tot_fam_mem!=
                                 age_summary$demographic_info_hh_members),]
  fam_mem_mismatch$issue= paste("Total family members is ",
                          fam_mem_mismatch$demographic_info_hh_members,
                          "whereas it is ", fam_mem_mismatch$tot_fam_mem,
                          "in age table")
  fam_mem_mismatch <- merge(fam_mem_mismatch, hh_info[,c("hh_unique_no",
                                       "location_name_interviewer")],
                    by.x="household_hh_unique_no",
```

```
                    by.y = "hh_unique_no", all.x=T)
  fam_mem_mismatch$cat <- "Mismatch in family member's number"
  fam_mem_mismatch$step <- "Step 1.2"
  names(fam_mem_mismatch)[1] <- "hh_unique_no"
  output1.2 <- rbind(output1.1, fam_mem_mismatch[,c("step", "cat", "hh_unique_no", "issue",
                                        "location_name_interviewer")])
}



#---------------------------------------------------------------------------------#
# 3.3 abnormal number of hh members
#     Input: age_summary from age.csv;
#     Output: list of households with abnornal number of hh members
#---------------------------------------------------------------------------------#

# select the problematic rows and put those on output
if (dim(age_summary[which(age_summary$tot_fam_mem==0|
                    age_summary$tot_fam_mem==1|
                    age_summary$tot_fam_mem> 15|
                    age_summary$third_gender>2),])[1] == 0){
  output1.3 <- rbind(output1.2, data.frame(step="Step 1.3",cat="Abnormal number of hh member",
                                 hh_unique_no=99999,
                                 issue= "No problem found",
                                 location_name_interviewer = "not applicable",
                                 stringsAsFactors = F))
} else {
  age_summary_prob <-  age_summary[which(age_summary$tot_fam_mem==0|
                                 age_summary$tot_fam_mem==1|
                                 age_summary$tot_fam_mem> 15|
```

```
                                age_summary$third_gender>2),]
  age_summary_prob$issue<- ifelse(age_summary_prob$tot_fam_mem>14,
                           paste("total family member is very high (",
                               age_summary_prob$tot_fam_mem,")"),
                           ifelse(age_summary_prob$tot_fam_mem==1,
                               "total family member only 1",
                               paste("third gender number strange (",
                                   age_summary_prob$third_gender,")")))
  age_summary_prob <- merge(age_summary_prob, hh_info[,c("hh_unique_no",
                                        "location_name_interviewer")],
                    by.x="household_hh_unique_no",
                    by.y = "hh_unique_no", all.x=T)
  age_summary_prob$cat <- "Abnormal number of hh member"
  age_summary_prob$step <- "Step 1.3"
  names(age_summary_prob)[1] <- "hh_unique_no"
  output1.3 <- rbind(output1.2, age_summary_prob[,c("step", "cat", "hh_unique_no", "issue",
                                    "location_name_interviewer")])
}



#-------------------------------------------------------------------------------#
# 1.4 Respondents age invalid (>80 yrs or <18 yrs)
# Input: "hh_info" from household.csv
# Output: list of hh where respondents were of >80 yrs or <18 yrs old
#-------------------------------------------------------------------------------#

#select the problem rows and put those in output file
if(dim(hh_info[which(hh_info$demographic_info_age_respondent>80|
                hh_info$demographic_info_age_respondent<18),])[1]==0){
```

```
  output1.4 <- rbind(output1.3, data.frame(step="Step 1.4",
                                           cat="Validity of respondent's age",
                                           issue="no problem found",
                                           hh_unique_no = 99999,
                                           location_name_interviewer="Not applicable"))
} else {
  respondent_age <- hh_info[which(hh_info$demographic_info_age_respondent>80|
                                    hh_info$demographic_info_age_respondent<18),]
  respondent_age$issue <- paste("respondent's age is",
                                respondent_age$demographic_info_age_respondent, sep = " ")
  respondent_age$step="Step 1.4"
  respondent_age$cat="Validity of respondent's age"
  names(respondent_age)[1] <- "hh_unique_no"
  output1.4 <- rbind(output1.3, respondent_age[,c("step", "cat", "hh_unique_no", "issue",
                                                  "location_name_interviewer")])
  }

#-------------------------------------------------------------------------------#
# 2 Inconsistency regarding primary forest product collection;
# 2.1 Each HH collected more than 5 tons per year collected but not sold??
#-------------------------------------------------------------------------------#
for_prodt_prob <- for_prodt

# select the problematic rows and put those on output
if (dim(for_prodt_prob[which(for_prodt$quantity_collection>5000 &
                               for_prodt$sold_product==0),])[1] == 0){
  output2.1 <- rbind(output1.4, data.frame(step="Step 2.1",cat="Inconsistency in collection amount",
                                           hh_unique_no=99999,
                                           issue="No problem found",
```

```
                        location_name_interviewer = "not applicable",
                        stringsAsFactors = F))
  } else {
    for_prodt_prob <-  for_prodt_prob[which(for_prodt$quantity_collection>5000 &
                                for_prodt$sold_product==0),]
    for_prodt_prob$issue= paste(">5 tons of product",(for_prodt_prob$forest_products_label),
                                 "collected but not sold")
    for_prodt_prob <- merge(for_prodt_prob, hh_info[,c("hh_unique_no",
                                    "location_name_interviewer")],
                    by.x="household_hh_unique_no",
                    by.y = "hh_unique_no", all.x=T)
    for_prodt_prob$step <-"Step 2.1"
    for_prodt_prob$cat <- "Inconsistency in collection amount"
    names(for_prodt_prob)[1] <- "hh_unique_no"
    output2.1 <- rbind(output1.4, for_prodt_prob[,c("step", "cat", "hh_unique_no",
                                "issue", "location_name_interviewer")])
    }

#-------------------------------------------------------------------------------#
# 2.2 data missing: forest product sold
# input: primary product collection; househod info
# output: list of hh missing primary forest product sold data
#-------------------------------------------------------------------------------#
# select the problematic rows and put those on output
if (dim(for_prodt[which(is.na(for_prodt$sold_product)),])[1] == 0){
  output2.2 <- rbind(output2.1, data.frame(step="Step 2.2",cat="Data missing: forest product sold",
                                hh_unique_no=99999,
                                issue="No problem found",
                                location_name_interviewer = "not applicable",
```

```
                                stringsAsFactors = F))
} else {
  for_prodt_miss <-  for_prodt[which(is.na(for_prodt$sold_product)),]
  for_prodt_miss$issue= "product sold and related data missing"
  for_prodt_miss <- merge(for_prodt_miss, hh_info[,c("hh_unique_no",
                                        "location_name_interviewer")],
                          by.x="household_hh_unique_no",
                          by.y = "hh_unique_no", all.x=T)
  for_prodt_miss$step <-"Step 2.2"
  for_prodt_miss$cat <- "Data missing: forest product sold"
  names(for_prodt_miss)[1] <- "hh_unique_no"
  output2.2 <- rbind(output2.1, for_prodt_miss[,c("step", "cat", "hh_unique_no",
                                      "issue", "location_name_interviewer")])
}

#-------------------------------------------------------------------------------#
# 2.3 data missing: LCC (location_tree_forests)) data missing
# Input: primary product collection
# OUtput: list of hh missing product collection LLC data
#-------------------------------------------------------------------------------#

# select the problematic rows (missing LCC data) and put those on output
if (dim(for_prodt[which(for_prodt$location_tree_forests_label.1.==""),])[1] == 0){
  output2.3 <- rbind(output2.2, data.frame(step="Step 2.2",cat="LCC data missing",
                                hh_unique_no=99999,
                                issue="All LCC are recorded",
                                location_name_interviewer = "not applicable",
                                stringsAsFactors = F))
} else {
```

```
 prodt_LCC_miss <- for_prodt[which(for_prodt$location_tree_forests_label.1.==""),]

 prodt_LCC_miss$issue= "LCC (location_tree_forest) of collected product missing"
 prodt_LCC_miss <- merge(prodt_LCC_miss, hh_info[,c("hh_unique_no",
                                    "location_name_interviewer")],
                 by.x="household_hh_unique_no",
                 by.y = "hh_unique_no", all.x=T)
 prodt_LCC_miss$step="Step 2.3"
 prodt_LCC_miss$cat= "LCC data missing"
 names(prodt_LCC_miss)[1] <- "hh_unique_no"
 output2.3 <- rbind(output2.2, prodt_LCC_miss[,c("step", "cat", "hh_unique_no",
                                  "issue", "location_name_interviewer")])
}

#-------------------------------------------------------------------------------#
# 2.4 Product sold unit price missing
# Input: primary product data
# Output: list of hh ID for which unit price of sold product is missed
#-------------------------------------------------------------------------------#
pri_prdt<- for_prodt

# select the problematic rows (missing LCC data) and put those on output
if(dim(pri_prdt[which(pri_prdt$qnty_sold== 1 &
                is.na(pri_prdt$unit_price)),])[1]==0){
 output2.4 <- rbind(output2.3, data.frame(step="Step 2.4",cat="Product price missing",
                            hh_unique_no=99999,
                            issue="no problem found",
                            location_name_interviewer = "not applicable",
                            stringsAsFactors = F))
```

```
}else{
 sold_price_miss <- pri_prdt[which(pri_prdt$qnty_sold== 1 &
                            is.na(pri_prdt$unit_price)),]
 sold_price_miss <- merge(sold_price_miss, hh_info[,c("hh_unique_no",
                                    "location_name_interviewer")],
                  by.x="household_hh_unique_no",
                  by.y="hh_unique_no",
                  all.x=TRUE)
 sold_price_miss$issue <- "primary product sold but unit price is missing"
 sold_price_miss$step <- "Step 2.4"
 sold_price_miss$cat <- "Product price missing"
 names(sold_price_miss)[1] <- "hh_unique_no"
 output2.4 <- rbind(output2.3, sold_price_miss[,c("step", "cat", "hh_unique_no",
                                  "issue", "location_name_interviewer")])
 }



#-------------------------------------------------------------------------------#
# 2.5 Primary product sold price is abnormal
# Input: primary forest product
# Output: List of hh for which abnormal unit price of sold product is reported
#-------------------------------------------------------------------------------#
pri_prdt<- for_prodt

# select the problematic rows (missing LCC data) and put those on output
if(dim(pri_prdt[which(pri_prdt$unit_price>1000),])[1]==0){
 output2.5 <- rbind(output2.4, data.frame(step="Step 2.5", cat="Product sold price missing",
                                  hh_unique_no = 99999,
```

```
                                issue = "no problem found",
                                location_name_interviewer = "not applicable"))
}else{
  price_abnormal <- pri_prdt[which(pri_prdt$unit_price>1000),]
  price_abnormal$issue <- "primary product sold unit price is abnormal (>1000 TK)"
  price_abnormal <- merge(price_abnormal, hh_info[,c("hh_unique_no",
                                        "location_name_interviewer")],
                  by.x="household_hh_unique_no",
                  by.y="hh_unique_no",
                  all.x=T)
  price_abnormal$step <- "Step 2.5"
  price_abnormal$cat <- "Product sold price missing"
  names(price_abnormal)[1] <- "hh_unique_no"
  output2.5 <- rbind(output2.4, price_abnormal[,c("step", "cat", "hh_unique_no",
                                      "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------------#
# 2.6 Product quantity collection very high (>10000 kg or no or cft)
# Input: primary product collection
# Output: list of hh having very abnormal quantity of product collected
#-------------------------------------------------------------------------------#
for_prodt <- merge(for_prodt, hh_info[,c("hh_unique_no",
                                "location_zone")],
              by.x="household_hh_unique_no",
              by.y = "hh_unique_no", all.x=T)
```

```
ggplot(for_prodt, aes(x=location_zone, y=quantity_collection))+
  geom_boxplot()+
  xlab("Forest zones")+ylab("Quantity of product collected (kg)")+
  ggtitle("Quanity of all products collected from different zones")+
  theme(plot.title = element_text(hjust=0.5))

ggplot(for_prodt[for_prodt$quantity_collection<10000,], aes(x=location_zone, y=quantity_collection))+
  geom_boxplot()+
  xlab("Forest zones")+ylab("Quantity of product collected (kg)")+
  ggtitle("Quanity of all products collected from different zones")+
  theme(plot.title = element_text(hjust=0.5))

#graph quality of timber
ggplot(for_prodt[for_prodt$forest_products==1 & for_prodt$quantity_collection<3000,],
       aes(x=location_zone, y=quantity_collection))+
  geom_boxplot()+
  xlab("Forest zones")+ylab("Quantity of timber collected (cft)")+
  ggtitle("Quanity of timber collected from different zones")+
  theme(plot.title = element_text(hjust=0.5))

#graph quality of timber
ggplot(for_prodt[for_prodt$forest_products==19,],
       aes(x=location_zone, y=quantity_collection))+
  geom_boxplot()+
  xlab("Forest zones")+ylab("Quantity of fuelwood collected (kg)")+
  ggtitle("Quanity of fuelwood collected from different zones")+
  theme(plot.title = element_text(hjust=0.5))
```

```
#graph quantity of timber
ggplot(for_prodt[for_prodt$forest_products==19,],
      aes(x=location_tree_forests.1., y=quantity_collection))+
  geom_boxplot()+
  xlab("Forest zones")+ylab("Quantity of fuelwood collected (kg)")+
  ggtitle("Quanity of fuelwood collected from different zones")+
  theme(plot.title = element_text(hjust=0.5))

for_prodt <- merge(for_prodt,
                   locations[,c("level1_code", "distname")],
                   by.y="level1_code",
                   by.x="household_hh_unique_no",
                   all.x=T)

ggplot(for_prodt[for_prodt$forest_products==19 &
                   for_prodt$distname=="Cox'S Bazar",],
      aes(x=location_tree_forests.1., y=quantity_collection))+
  geom_boxplot()+
  xlab("Forest zones")+ylab("Quantity of fuelwood collected (kg)")+
  ggtitle("Quanity of fuelwood collected from Cox's Bazar")+
  theme(plot.title = element_text(hjust=0.5))



ggplot(for_prodt[for_prodt$quantity_collection<10000,],
      aes(x=location_tree_forests.1., y=quantity_collection))+
  geom_boxplot()+
  xlab("Land Cover Class")+ylab("Quantity of product collected (kg)")+
  ggtitle("Quanity of all products collected from different LCC")+
```

```
  theme(plot.title = element_text(hjust=0.5))



#list the problematic rows and put those in output
if(dim(for_prodt[which(for_prodt$quantity_collection>=10000),])[1]==0){
  output2.6 <- rbind(output2.5, data.frame(step="Step 2.6",
                                           cat= "Product quantity high",
                                           hh_unique_no=99999,
                                           issue="No problem found",
                                           location_name_interviewer="Not applicable",
                                           stringsAsFactors = F))
}else{
  for_prodt_high <- for_prodt[which(for_prodt$quantity_collection>10000),]
  for_prodt_high$issue <- paste(for_prodt_high$forest_products_qualifier,
                                "collection is very high (",
                                for_prodt_high$quantity_collection,")", sep = " ")
  for_prodt_high <- merge(for_prodt_high, hh_info[,c("hh_unique_no",
                                                     "location_name_interviewer")],
                          by.x="household_hh_unique_no",
                          by.y="hh_unique_no",
                          all.x=TRUE)
  for_prodt_high$step <- "Step 2.6"
  for_prodt_high$cat <- "Product quantity high"
  names(for_prodt_high)[1] <- "hh_unique_no"
  output2.6 <- rbind(output2.5, for_prodt_high[,c("step", "cat",
                                                  "hh_unique_no",
                                                  "issue",
                                                  "location_name_interviewer")])
```

```
}

#-------------------------------------------------------------------------#
# 2.7 inconsistency in primary product collected Vs sold
# Input: primary product collection, household info
# Output: list of hh associated very high amount of product collection
#-------------------------------------------------------------------------#


#list the problematic rows and incorporate them in output
if(dim(for_prodt[which(for_prodt$quantity_collection <
                  for_prodt$qnty_sold),])[1]==0){
  output2.7 <- rbind(output2.6, data.frame(step="Step 2.7",
                                cat="inconsistency in product collection",
                                hh_unique_no=99999,
                                issue="No problems found",
                                location_name_interviewer="Not applicable",
                                stringsAsFactors = F))
}else{
  for_prodt_sold <- for_prodt[which(for_prodt$quantity_collection <
                               for_prodt$qnty_sold),]
  for_prodt_sold$issue <- paste("product quanity collection ",
                          for_prodt_sold$quantity_collection,
                          " kg but sold",
                          for_prodt_sold$qnty_sold, " kg")


  for_prodt_sold <- merge(for_prodt_sold, hh_info[,c("hh_unique_no",
                                        "location_name_interviewer")],
```

```
                    by.x="household_hh_unique_no",
                    by.y = "hh_unique_no", all.x=T)
  for_prodt_sold$step <- "Step 2.7"
  for_prodt_sold$cat <- "inconsistency in product collection"
  names(for_prodt_sold)[1]<- "hh_unique_no"
  output2.7 <- rbind(output2.6, for_prodt_sold[,c("step", "cat", "hh_unique_no",
                                    "issue", "location_name_interviewer")])
}



ggplot(for_prodt, aes(x=quantity_collection, y=qnty_sold))+
  geom_point()+
  xlab("Quanity of forest product collected (kg)")+ylab("Quantity of product sold (kg)")+
  ggtitle("Quanity of product collected Vs sold")+
  theme(plot.title = element_text(hjust=0.5))



#-------------------------------------------------------------------------------#
# 2.8 Collection of product from invalid LCC
# Input: land cover area, primary product
# output: list of hh where location of collection (LCC) is seemed invalid
#-------------------------------------------------------------------------------#
pri_prdt <- for_prodt

#Keep necessary columns only
pri_prdt <- pri_prdt[,c("household_hh_unique_no", "location_tree_forests.1.",
                    "location_zone", "quantity_collection",
                    "forest_products_qualifier")]
```

```
pri_prdt <- unique(pri_prdt)


#add LCC area per zone
#turn columns of lc area by zone into rows
require(reshape2)
lc_area_molted <- lc_area[,-c(2:3)]
lc_area_molted <- melt(lc_area_molted, "Land_cover_classes")
lc_area_molted$location_zone <- ifelse(lc_area_molted$variable=="lc_village_ha",
                                "Villages",
                                ifelse(lc_area_molted$variable=="lc_sal_ha",
                                     "Sal",
                                     ifelse(lc_area_molted$variable=="lc_sundarbans_ha",
                                          "Sundarbans",
                                          ifelse(lc_area_molted$variable=="lc_hill_ha",
                                               "Hill", "Coastal"))))

names(lc_area_molted)[3] <- "area_m2"
lc_area_molted$area_ha <- lc_area_molted$area_m2/10000
lc_area_molted[,2:3] <- NULL

#convert S and ShF LCC into ShT LCC to fit data available
pri_prdt$location_tree_forests.1. <- ifelse(pri_prdt$location_tree_forests.1.=="S", "ShT",
                                  ifelse(pri_prdt$location_tree_forests.1.=="ShF", "ShT",
                                       pri_prdt$location_tree_forests.1.))



pri_prdt <- merge(pri_prdt[which(pri_prdt$location_tree_forests.1.!=""),], lc_area_molted,
               by.x = c("location_zone", "location_tree_forests.1."),
```

```
            by.y = c("location_zone", "Land_cover_classes"),
            all.x=TRUE)



#select hh and cases where invalid LCC was selected and put those in output file
if(dim(pri_prdt[which(pri_prdt$quantity_collection>0 &
                pri_prdt$area_ha==0),])[1]==0){
  output2.8 <- rbind(output2.7, data.frame(step="Step 2.8",
                                cat="Invalid LCC reported considering zone",
                                hh_unique_no = 99999,
                                issue= "No problem found",
                                location_name_interviewer="Not applicable"))
}else{
  lcc_invalid <- pri_prdt[which(pri_prdt$quantity_collection>0 &
                          pri_prdt$area_ha==0),]

  lcc_invalid$issue <- paste("Product collected from LCC", lcc_invalid$location_tree_forests.1.,
                     "in", lcc_invalid$location_zone, "but no area is reported",
                     sep=" ")
  lcc_invalid <- merge(lcc_invalid, hh_info[,c("hh_unique_no", "location_name_interviewer")],
                 by.x="household_hh_unique_no",
                 by.y="hh_unique_no",
                 all.x=T)
  lcc_invalid$step <- "step 2.8"
  lcc_invalid$cat <- "Invalid LCC reported considering zone"
  names(lcc_invalid)[1]<- "hh_unique_no"
  output2.8 <- rbind(output2.7, lcc_invalid[,c("step", "cat", "hh_unique_no",
                                "issue", "location_name_interviewer")])
}
```

```
#-----------------------------------------------------------------------------#
# 3 Conflict
# 3.1 Conflict mentioned but conflict frequency and related other info is missed
# Input: conflict tree forest data;
# Output: List of hh id for which conflict frequency and related info is missed
#-----------------------------------------------------------------------------#

#Identify problematic rows and add to output table
if(dim(conflict[which(!is.na(conflict$type_conflict) &
                (is.na(conflict$frequency_conflict)|
                 is.na(conflict$conflict_manage.1.))),])[1]==0){
 output3.1<- rbind(output2.8, data.frame(step="Step 3.1",
                                cat="conflict frequency and related info is missed",
                                hh_unique_no=99999,
                                issue= "No problem found",
                                location_name_interviewer="Not applicable",
                                stringsAsFactors = F))
 }else{
 conflict_info_miss <- conflict[which(!is.na(conflict$type_conflict) &
                               (is.na(conflict$frequency_conflict)|
                                  is.na(conflict$conflict_manage.1.))),]
 conflict_info_miss <- merge(conflict_info_miss, hh_info[,c("hh_unique_no",
                                          "location_name_interviewer")],
              by.x="household_hh_unique_no",
              by.y="hh_unique_no",
              all.x=T)
```

```
 conflict_info_miss$issue <- "Conflict frequency or other related info missed"
 conflict_info_miss$cat <- "conflict frequency and related info is missed"
 conflict_info_miss$step <- "Step 3.1"
 names(conflict_info_miss)[1]<- "hh_unique_no"
 output3.1 <- rbind(output2.8, conflict_info_miss[,c("step", "cat",
                                    "hh_unique_no",
                                    "issue",
                                    "location_name_interviewer")])
 }

#---------------------------------------------------------------------------#
# 3.2 Total number of Conflict per hh per year
# Input: conflict tree forest data;
# Output: List of hh id for which number of conflict is exceptioinally high
#---------------------------------------------------------------------------#

conflict_no <- aggregate(frequency_conflict~type_conflict+household_hh_unique_no,
                  data=conflict, FUN=sum)

#no of conflict irrespective of conflict type
conflict_no <- aggregate(frequency_conflict~household_hh_unique_no,
                  data=conflict_no, FUN=sum)

# more than 5 conflict per hh seems exceptionally high need to be checked,
# list the hh facing more than 5 conflict/year and put those in output

if(dim(conflict_no[which(conflict_no$frequency_conflict>5),])[1]==0){
 output3.2 <- rbind(output3.1, data.frame(step="Step 3.2",
                           cat="Conflict number
```

```
                        is exceptionally high",
                        issue="No problems found",
                        hh_unique_no = 99999,
                        location_name_interviewer="Not
                        Applicable",
                        stringsAsFactors = F))
 }else{
 conflict_no_high <- conflict_no[which(conflict_no$frequency_conflict>5),]
 conflict_no_high$issue <- paste("total number of conflict is
                    exceptionally high",
                    conflict_no_high$frequency_conflict)
 conflict_no_high$cat <- "Conflict number is exceptionally high"
 conflict_no_high$step <- "Step 3.2"
 conflict_no_high <- merge(conflict_no_high, hh_info[,c("hh_unique_no",
                                "location_name_interviewer")],
                by.x="household_hh_unique_no",
                by.y="hh_unique_no",
                all.x=T)
 names(conflict_no_high)[1]<-"hh_unique_no"
 output3.2 <- rbind(output3.1, conflict_no_high[,c("step", "cat", "hh_unique_no",
                                "issue",
                                "location_name_interviewer")])
}



#-------------------------------------------------------------------------#
# 4. Cyclone
# 4.1 No damage but economic loss mentioned
# Input: cyclone damage data;
```

```
# Output: List of hh id where hh_damage is 0 but  economic loss is mentioned
#-----------------------------------------------------------------------------#
cyclone_dam <- cyclone


if(dim(cyclone_dam[which(cyclone_dam$hh_damaged==0 &
                         cyclone_dam$hh_economic_loss>0),])[1]==0){
  output4.1 <- rbind(output3.2, data.frame(step="Step 4.1",
                                           cat="damage and economic loss
                                           info inconsistant",
                                           issue="No problem found",
                                           hh_unique_no = 99999,
                                           location_name_interviewer="Not applicable",
                                           stringsAsFactors = F))
}else{
  cyclone_no_dam <- cyclone_dam[which(cyclone_dam$hh_damaged==0 &
                                      cyclone_dam$hh_economic_loss>0),]
  cyclone_no_dam$step <- "Step 4.1"
  cyclone_no_dam$cat <- "damage and economic loss info inconsistant"
  cyclone_no_dam$issue <- paste("hh_damage is 0 but economic loss mentioned BDT",
                                cyclone_no_dam$hh_economic_loss)
  cyclone_no_dam <- merge(cyclone_no_dam, hh_info[,c("hh_unique_no",
                                                     "location_name_interviewer")],
                          by.x="household_hh_unique_no",
                          by.y="hh_unique_no",
                          all.x=T)
  names(cyclone_no_dam)[1]<- "hh_unique_no"
  output4.1 <- rbind(output3.2, cyclone_no_dam[,c("step", "cat", "hh_unique_no",
                                                  "issue",
                                                  "location_name_interviewer")])
```

```
 }



#-------------------------------------------------------------------------#
# 4.2 hh_damage zero but economic loss is NA instead of zero
# Input: cyclone damage data;
# Output: List of hh where hh_damage is 0 but economic loss is NA instead of 0
#-------------------------------------------------------------------------#

if(dim(cyclone_dam[which(cyclone_dam$hh_damaged==0 &
                  (is.na(cyclone_dam$hh_economic_loss))),])[1]==0){
 output4.2 <- rbind(output4.1, data.frame(step="Step 4.2",
                                 cat="damage and economic loss
                                 info inconsistant",
                                 issue="No problem found",
                                 hh_unique_no = 99999,
                                 location_name_interviewer="Not applicable",
                                 stringsAsFactors = F))
 }else{
 cyclone_dam_incons <- cyclone_dam[which(cyclone_dam$hh_damaged==0 &
                               (is.na(cyclone_dam$hh_economic_loss))),]
 cyclone_dam_incons$step <- "Step 4.2"
 cyclone_dam_incons$cat <- "damage and economic loss info inconsistant"
 cyclone_dam_incons$issue <- "hh_damage is 0 but economic loss NA instead of 0"
 cyclone_dam_incons <- merge(cyclone_dam_incons, hh_info[,c("hh_unique_no",
                                      "location_name_interviewer")],
                  by.x="household_hh_unique_no",
```

```
                    by.y="hh_unique_no",
                    all.x=T)
 names(cyclone_dam_incons)[1]<- "hh_unique_no"
 output4.2 <- rbind(output4.1, cyclone_dam_incons[,c("step", "cat", "hh_unique_no",
                                    "issue",
                                    "location_name_interviewer")])
 }


#-------------------------------------------------------------------------------#
# 4.3 hh_damage mentioned but economic loss is absent
# Input: cyclone damage data;
# Output: List of hh id where hh_damage mentioned but economic loss is absent
#-------------------------------------------------------------------------------#

if(dim(cyclone_dam[which(cyclone_dam$hh_damaged>0 &
                   (is.na(cyclone_dam$hh_economic_loss)|
                    cyclone_dam$hh_economic_loss==0)),])[1]==0){
 output4.3 <- rbind(output4.2, data.frame(step="Step 4.3",
                                  cat="damage and economic loss
                                  info inconsistant",
                                  issue="No problem found",
                                  hh_unique_no = 99999,
                                  location_name_interviewer="Not applicable",
                                  stringsAsFactors = F))
 }else{
 cyclone_dam_incons <- cyclone_dam[which(cyclone_dam$hh_damaged>0 &
                                  (is.na(cyclone_dam$hh_economic_loss)|
                                     cyclone_dam$hh_economic_loss==0)),]
```

```
 cyclone_dam_incons$step <- "Step 4.3"
 cyclone_dam_incons$cat <- "damage and economic loss info inconsistant"
 cyclone_dam_incons$issue <- "hh_damage mentioned but economic loss is missing"
 cyclone_dam_incons <- merge(cyclone_dam_incons, hh_info[,c("hh_unique_no",
                                        "location_name_interviewer")],
                  by.x="household_hh_unique_no",
                  by.y="hh_unique_no",
                  all.x=T)
 names(cyclone_dam_incons)[1]<- "hh_unique_no"
 output4.3 <- rbind(output4.2, cyclone_dam_incons[,c("step", "cat", "hh_unique_no",
                                    "issue",
                                    "location_name_interviewer")])
 }



#-------------------------------------------------------------------------#
# 4.4 Double entry of same hh_damage for single hh
# Input: cyclone damage data;
# Output: List of hh id hh_position entered twice for single type of damage
#-------------------------------------------------------------------------#

cyclone_dam <- cyclone
cyclone_dam$count <- 1
cyclone_dam <- aggregate(count~household_hh_unique_no+hh_damaged,
                 data=cyclone_dam, FUN=sum)
if(dim(cyclone_dam[which(cyclone_dam$count>1),])[1]==0){
 output4.4 <- rbind(output4.3, data.frame(step="Step 4.4",
                          cat="unnecessary double entry of
                          same info",
```

```
                        issue="No problem found",
                        hh_unique_no = 99999,
                        location_name_interviewer="Not applicable",
                        stringsAsFactors = F))
 }else{
 cyclone_dam_incons <- cyclone_dam[which(cyclone_dam$count>1),]
 cyclone_dam_incons$step <- "Step 4.4"
 cyclone_dam_incons$cat <- "unnecessary double entry of same info"
 cyclone_dam_incons$issue <- paste("hh_damage number", cyclone_dam_incons$hh_damaged,
                    "entered", cyclone_dam_incons$count, "times")
 cyclone_dam_incons <- merge(cyclone_dam_incons, hh_info[,c("hh_unique_no",
                                "location_name_interviewer")],
                  by.x="household_hh_unique_no",
                  by.y="hh_unique_no",
                  all.x=T)
 names(cyclone_dam_incons)[1]<- "hh_unique_no"
 output4.4 <- rbind(output4.3, cyclone_dam_incons[,c("step", "cat", "hh_unique_no",
                                "issue",
                                "location_name_interviewer")])
 }


#-------------------------------------------------------------------------#
# 4.5 Cyclone damage amount (BDT) > 500 000 TK per damage type per hh
# Input: cyclone damage data;
# Output: List of hh id where economic loss >500000 per damage type per hh
#-------------------------------------------------------------------------#

cyclone_dam <- cyclone
cyclone_dam[order(-cyclone_dam$hh_economic_loss),][1:10,]
```

```
#add hh_damage label
cyclone_dam$hh_damaged_qualifier <- ifelse(cyclone_dam$hh_damaged==0,
                                "No loss or damages",
                                ifelse(cyclone_dam$hh_damaged==1,
                                    "Loss of/damage to house",
                                    ifelse(cyclone_dam$hh_damaged==2,
                                        "Loss of agricultural crops",
                                        ifelse(cyclone_dam$hh_damaged==3,
                                            "Loss of/damage to trees",
                                            ifelse(cyclone_dam$hh_damaged==4,
                                                "Loss of/damage to fish ponds",
                                                ifelse(cyclone_dam$hh_damaged==5,
                                                    "Loss of/damage to shrimp farms",
                                                    ifelse(cyclone_dam$hh_damaged==6,
                                                        "Loss of livestock",
                                                        "Other losses or damages")))))))


if(dim(cyclone_dam[which(cyclone_dam$hh_economic_loss>500000),])[1]==0){
 output4.5 <- rbind(output4.4, data.frame(step="Step 4.5",
                                cat="economic loss >500000TK
                                per damage type per hh",
                                issue="No problem found",
                                hh_unique_no = 99999,
                                location_name_interviewer="Not applicable",
                                stringsAsFactors = F))
 }else{
 cyclone_dam_high <- cyclone_dam[which(cyclone_dam$hh_economic_loss>500000),]
```

```
 cyclone_dam_high$step <- "Step 4.5"
 cyclone_dam_high$cat <- "economic loss >500000TK per damage type per hh"
  cyclone_dam_high$issue <- paste(cyclone_dam_high$hh_damaged_qualifier,
                          "is >500000 TK (",
                         cyclone_dam_high$hh_economic_loss, ")")
 cyclone_dam_high <- merge(cyclone_dam_high, hh_info[,c("hh_unique_no",
                                      "location_name_interviewer")],
                    by.x="household_hh_unique_no",
                    by.y="hh_unique_no",
                    all.x=T)
 names(cyclone_dam_high)[1]<- "hh_unique_no"
 output4.5 <- rbind(output4.4, cyclone_dam_high[,c("step", "cat", "hh_unique_no",
                                    "issue",
                                    "location_name_interviewer")])
 }



#-------------------------------------------------------------------------#
# 4.6 Total damage amount (BDT) per hh is exceptionally high
# Input: cyclone damage data;
# Output: List of hh id where economic loss for hh_damage is exceptionally high
#-------------------------------------------------------------------------#

cyclone_dam <- cyclone
cyclone_dam <- aggregate(hh_economic_loss~household_hh_unique_no,
                  data=cyclone_dam, FUN=sum)

cyclone_dam[order(-cyclone_dam$hh_economic_loss),][1:10,]
```

```
#economic loss > 1500000 TK seemed exceptionally high need to be checked
if(dim(cyclone_dam[which(cyclone_dam$hh_economic_loss>1500000),])[1]==0){
 output4.6 <- rbind(output4.5, data.frame(step="Step 4.6",
                                          cat="economic loss excetionally high",
                                          issue="No problem found",
                                          hh_unique_no = 99999,
                                          location_name_interviewer="Not applicable",
                                          stringsAsFactors = F))
 }else{
 cyclone_dam_high <- cyclone_dam[which(cyclone_dam$hh_economic_loss>1500000),]
 cyclone_dam_high$step <- "Step 4.6"
 cyclone_dam_high$cat <- "economic loss excetionally high"
 cyclone_dam_high$issue <- paste("economic loss very high (",
                                 cyclone_dam_high$hh_economic_loss,
                                  "TK)", sep = "")
 cyclone_dam_high <- merge(cyclone_dam_high, hh_info[,c("hh_unique_no",
                                                        "location_name_interviewer")],
                           by.x="household_hh_unique_no",
                           by.y="hh_unique_no",
                           all.x=T)
 names(cyclone_dam_high)[1]<- "hh_unique_no"
 output4.6 <- rbind(output4.5, cyclone_dam_high[,c("step", "cat", "hh_unique_no",
                                                   "issue",
                                                   "location_name_interviewer")])
 }
```

```
#-----------------------------------------------------------------------------#
# 5 Potable water collection
# 5.1 data missing in water collection table
# 5.1.1 Water collection location missing
# 5.1.2 hh_water_collect information missing
# 5.1.3 months_water data missing
# 5.1.4 hours_water data missing
# 5.1.5 amount_water data missing
# 5.1.6 cost_water_trans data missing
# Input: water_collection.csv
# Output: List of hh id with data missing in water collection table
#-----------------------------------------------------------------------------#
wat_data_miss <- wat_collct

if(dim(wat_data_miss[which(wat_data_miss$location_water_sources.1.==""|
                           is.na(wat_data_miss$hh_water_collect)|
                           is.na(wat_data_miss$months_water)|
                           is.na(wat_data_miss$hours_water)|
                           is.na(wat_data_miss$amount_water)|
                           is.na(wat_data_miss$cost_water_trans)),])[1]==0){
output5.1 <- rbind(output4.6,
            data.frame(step= "Step 5.1", cat="data missing in water collection table",
                       hh_unique_no= 99999,
                       issue="No Problem Found",
                       location_name_interviewer="Not Applicable"))
 }else{
   wat_data_miss <- wat_data_miss[which(wat_data_miss$location_water_sources.1.==""|
                    is.na(wat_data_miss$hh_water_collect)|
                    is.na(wat_data_miss$months_water)|
```

```
                is.na(wat_data_miss$hours_water)|
                is.na(wat_data_miss$amount_water)|
                is.na(wat_data_miss$cost_water_trans)),]
  wat_data_miss$issue <-paste(ifelse(wat_data_miss$location_water_sources.1.=="",
                        "Water collection location missing",
                        ifelse(is.na(wat_data_miss$hh_water_collect),
                            "hh member collects water data missing",
                            ifelse(is.na(wat_data_miss$months_water),
                                "water collection months data missing",
                                ifelse(is.na(wat_data_miss$hours_water),
                                    "water collection hours data missing",
                                    ifelse(is.na(wat_data_miss$amount_water),
                                        "water collected amount data missing",
                                        "water transport cost data missing"))))))
 wat_data_miss$step <- "step 5.1"
 wat_data_miss$cat <- "data missing in water collection table"
 wat_data_miss <- merge(wat_data_miss, hh_info[,c("hh_unique_no",
                                "location_name_interviewer")],
                by.x="household_hh_unique_no",
                by.y = "hh_unique_no",
                all.x = T)
 names(wat_data_miss)[1]<- "hh_unique_no"
 output5.1<- rbind(output4.6, wat_data_miss[,c("step", "cat", "hh_unique_no",
                                "issue",
                                "location_name_interviewer")])
}
```

```
#-----------------------------------------------------------------------------#
# 5.2 water collection months inconsistent (more than 12 or less than 6)
# Input: water_collection.csv; household.csv
# Output: List of hh id where water is collected for > 12 months and
# water collection source is rainfall but collected for > 6 months
#-----------------------------------------------------------------------------#
wat_collct_incons <- wat_collct
if(dim(wat_collct_incons[which(wat_collct_incons$months_water>12|
                        (wat_collct_incons$sources_water==5 &
                         wat_collct_incons$months_water>6)),])[1]==0){
 output5.2 <- rbind(output5.1, data.frame(step="Step 5.2", cat= "Water collection
                                 months inconsistent",
                                 hh_unique_no = 99999,
                                 issue="No problem found",
                                 location_name_interviewer="Not applicable"))
 }else{
 wat_collct_incons <- wat_collct_incons[which(wat_collct_incons$months_water>12|
                                    (wat_collct_incons$sources_water==5 &
                                      wat_collct_incons$months_water>6)),]
 wat_collct_incons$step <- "Step 5.2"
 wat_collct_incons$cat <- "Water collection months inconsistent"
 wat_collct_incons$issue <- paste(ifelse(wat_collct_incons$months_water>12,
                                 "collects water for > 12 months",
                                 "water source is rain but collects
                                 for >6 months"))
 wat_collct_incons <- merge(wat_collct_incons, hh_info[,c("hh_unique_no",
                                          "location_name_interviewer")],
                     by.x="household_hh_unique_no",
                     by.y="hh_unique_no",
```

```
                    all.x=T)
 names(wat_collct_incons)[1] <- "hh_unique_no"
 output5.2 <- rbind(output5.1, wat_collct_incons[,c("step", "cat",
                                    "hh_unique_no",
                                    "issue",
                                    "location_name_interviewer")])
 }



#-------------------------------------------------------------------------#
# 5.3 Water collection amount less than 5 litre per day
# Input: water_collection.csv; household.csv
# Output: List of hh id where total water collection is
#       less than 2 litre/day/hh member
#-------------------------------------------------------------------------#
wat_collct_amount <- wat_collct
wat_collct_amount <- aggregate(amount_water~household_hh_unique_no,
                    data=wat_collct_amount, FUN=sum)

wat_collct_amount <-  merge(wat_collct_amount, hh_info[,c("hh_unique_no",
                                    "demographic_info_hh_members")],
                    by.x="household_hh_unique_no",
                    by.y="hh_unique_no",
                    all.x=T)
wat_collct_amount$wat_per_hh_member <- round((wat_collct_amount$amount_water/
                        wat_collct_amount$demographic_info_hh_members), 2)

#list the problematic rows and put those in the output table
```

```
if(dim(wat_collct_amount[which(wat_collct_amount$wat_per_hh_member<2),])[1]==0){
 output5.3 <- rbind(output5.2, data.frame(step = "step 5.3", cat="water collection
                                  amount inconsistent",
                                  hh_unique_no = 99999,
                                  issue="No problems found",
                                  location_name_interviewer="Not applicable"))
 }else{
 wat_collct_amount <- wat_collct_amount[which(wat_collct_amount$wat_per_hh_member<2),]
 wat_collct_amount$step <- "Step 5.3"
 wat_collct_amount$cat <- "water collection amount inconsistent"
 wat_collct_amount$issue <- paste("water collection amount per hh member is very less (",
                            wat_collct_amount$wat_per_hh_member, ") litre", sep = "")
 wat_collct_amount <- merge(wat_collct_amount, hh_info[,c("hh_unique_no",
                                        "location_name_interviewer")],
                    by.x = "household_hh_unique_no",
                    by.y = "hh_unique_no",
                    all.x=T)
 names(wat_collct_amount)[1]<- "hh_unique_no"
 output5.3<- rbind(output5.2, wat_collct_amount[,c("step", "cat",
                                    "hh_unique_no",
                                    "issue", "location_name_interviewer")])
 }


#-------------------------------------------------------------------------------#
# 5.4 Water collection time (hrs/day) is exceptionally high
# Input: water_collection.csv; household.csv
# Output: List of hh id where total water collection time is
#        exceptionally high (>/= 5 hrs/day)
#-------------------------------------------------------------------------------#
```

```
wat_collct_time <- wat_collct
wat_collct_time <- aggregate(hours_water~household_hh_unique_no,
                     data=wat_collct_time,
                     FUN=sum)

if(dim(wat_collct_time[which(wat_collct_time$hours_water>=5),])[1]==0){
  output5.4 <- rbind(output5.3, data.frame(step = "step 5.4", cat="water collection
                                time inconsistent",
                                hh_unique_no = 99999,
                                issue="No problems found",
                                location_name_interviewer="Not applicable"))
  }else{
  wat_collct_time <- wat_collct_time[which(wat_collct_time$hours_water>=5),]
  wat_collct_time$step <- "Step 5.4"
  wat_collct_time$cat <- "water collection time inconsistent"
  wat_collct_time$issue <- paste("water collection hours per day is
                          exceptionally high (",
                          wat_collct_time$hours_water, " hrs/day)", sep = "")
  wat_collct_time <- merge(wat_collct_time, hh_info[,c("hh_unique_no",
                                        "location_name_interviewer")],
                     by.x = "household_hh_unique_no",
                     by.y = "hh_unique_no",
                     all.x=T)
  names(wat_collct_time)[1]<- "hh_unique_no"
  output5.4<- rbind(output5.3, wat_collct_time[,c("step", "cat",
                                    "hh_unique_no",
                                    "issue", "location_name_interviewer")])
}
```

```
#-----------------------------------------------------------------------------#
# 5.5 Water transportation cost unusual (1-30 tk/month)
# Input: water_collection.csv; household.csv
# Output: List of hh id where water transportation cost is
#         unusual (1-30 tk/month)
#-----------------------------------------------------------------------------#
wat_collct_cost <- wat_collct
wat_collct_cost <- aggregate(cost_water_trans~household_hh_unique_no,
                     data=wat_collct_cost, FUN=sum)

if(dim(wat_collct_cost[which(wat_collct_cost$cost_water_trans>=1|
                     wat_collct_cost$cost_water_trans<=30),])[1]==0){
  output5.5 <- rbind(output5.4, data.frame(step = "step 5.5", cat="water collection
                                cost inconsistent",
                                hh_unique_no = 99999,
                                issue="No problems found",
                                location_name_interviewer="Not applicable"))
  }else{
    wat_collct_cost <- wat_collct_cost[which(wat_collct_cost$cost_water_trans>=1|
                                  wat_collct_cost$cost_water_trans<=30),]
    wat_collct_cost$step <- "Step 5.5"
    wat_collct_cost$cat <- "water collection cost inconsistent"
    wat_collct_cost$issue <- paste("water collection cost per month is unusual (",
                         wat_collct_cost$cost_water_trans, " tk/month)",
                         sep = "")
    wat_collct_cost <- merge(wat_collct_cost, hh_info[,c("hh_unique_no",
                                          "location_name_interviewer")],
                       by.x = "household_hh_unique_no",
```

```
                    by.y = "hh_unique_no",
                    all.x=T)
  names(wat_collct_cost)[1]<- "hh_unique_no"
  output5.5<- rbind(output5.4, wat_collct_time[,c("step", "cat",
                                    "hh_unique_no",
                                    "issue", "location_name_interviewer")])
  }



#------------------------------------------------------------------------------#
# 6 Water treatment
# 6.1 cost_water_treat cost data is NA
# Input: water_treatment.csv; household.csv
# Output: list of hh for which the water_treatment cost is NA
#------------------------------------------------------------------------------#
treat_data_miss <- wat_treat

if(dim(treat_data_miss[which(is.na(treat_data_miss$cost_water_treat)),])[1]==0){
 output6.1 <- rbind(output5.5, data.frame(step = "step 6.1", cat="water treatment
                                  cost is NA",
                                  hh_unique_no = 99999,
                                  issue="No problems found",
                                  location_name_interviewer="Not applicable"))
 }else{
 treat_data_miss <- treat_data_miss[which(is.na(treat_data_miss$cost_water_treat)),]
 treat_data_miss$step <- "Step 6.1"
 treat_data_miss$cat <- "water treatment cost is NA"
 treat_data_miss$issue <- "water treatment cost is NA"
```

```
 treat_data_miss <- merge(treat_data_miss, hh_info[,c("hh_unique_no",
                                    "location_name_interviewer")],
                  by.x = "household_hh_unique_no",
                  by.y = "hh_unique_no",
                  all.x=T)
 names(treat_data_miss)[1]<- "hh_unique_no"
 output6.1<- rbind(output5.5, treat_data_miss[,c("step", "cat",
                                  "hh_unique_no",
                                  "issue", "location_name_interviewer")])
 }



#-------------------------------------------------------------------------#
# 6.2 Water is treated but treatment cost is 0
# Input: water_treatment.csv
# Output: List of hh for which water is treated but treatment cost is 0
#-------------------------------------------------------------------------#
wat_treat_cost <- wat_treat

if(dim(wat_treat_cost[which(wat_treat_cost$type_water_treat>0 &
                  wat_treat_cost$cost_water_treat==0),])[1]==0){
 output6.2 <- rbind(output6.1, data.frame(step = "step 6.2", cat="water
                                  treatment cost inconsistent",
                                  hh_unique_no = 99999,
                                  issue="No problems found",
                                  location_name_interviewer="Not applicable"))
}else{
 wat_treat_cost <- wat_treat_cost[which(wat_treat_cost$type_water_treat>0 &
                                   wat_treat_cost$cost_water_treat==0),]
```

```
  wat_treat_cost$step <- "Step 6.2"
  wat_treat_cost$cat <- "water treatment cost inconsistent"
  wat_treat_cost$issue <- "water is treated but treatment cost is 0"
  wat_treat_cost <- merge(wat_treat_cost, hh_info[,c("hh_unique_no",
                                        "location_name_interviewer")],
                    by.x = "household_hh_unique_no",
                    by.y = "hh_unique_no",
                    all.x=T)
  names(wat_treat_cost)[1]<- "hh_unique_no"
  output6.2<- rbind(output6.1, wat_treat_cost[,c("step", "cat",
                                    "hh_unique_no",
                                    "issue",
                                    "location_name_interviewer")])
}



#-------------------------------------------------------------------------------#
# 6.3 Water treatment cost unrealistic
# Input: water_treatment.csv
# Output: List of hh for which water treatment cost >/= 10 TK/litre
#-------------------------------------------------------------------------------#

#water treatment cost per litre per jth hh
wat_treat_cost <- wat_treat[which(!is.na(wat_treat$cost_water_treat)),]
wat_treat_cost$treat_cost_per_ltr_tk <- wat_treat_cost$cost_water_treat/10

if(dim(wat_treat_cost[wat_treat_cost$treat_cost_per_ltr_tk>=10,])[1]==0){
output6.3 <- rbind(output6.2, data.frame(step = "step 6.3", cat="water
                                  treatment cost much high",
```

```
                                hh_unique_no = 99999,
                                issue="No problems found",
                                location_name_interviewer="Not applicable"))
}else{
  wat_treat_cost <- wat_treat_cost[wat_treat_cost$treat_cost_per_ltr_tk>=10,]
  wat_treat_cost$step <- "Step 6.3"
  wat_treat_cost$cat <- "water treatment cost much high"
  wat_treat_cost$issue <- paste("water is treatment cost is",
                          wat_treat_cost$treat_cost_per_ltr_tk, "tk per litre")
  wat_treat_cost <- merge(wat_treat_cost, hh_info[,c("hh_unique_no",
                                      "location_name_interviewer")],
                  by.x = "household_hh_unique_no",
                  by.y = "hh_unique_no",
                  all.x=T)
  names(wat_treat_cost)[1]<- "hh_unique_no"
  output6.3<- rbind(output6.2, wat_treat_cost[,c("step", "cat",
                                  "hh_unique_no",
                                  "issue",
                                  "location_name_interviewer")])
}




#-------------------------------------------------------------------------#
# 7 forest product entry fee
# 7.1 missing data in pass_permit_product, number_collection and
#     amount_entry (amount of pass permit per trip for each product)
# input: entry_fee_product_srf.csv
```

```
# output: list of hh where data is missing
#------------------------------------------------------------------------#

permit_amount <- entry_fee

if(dim(permit_amount[which(is.na(permit_amount$pass_permit_product)|
                    is.na(permit_amount$number_collection)|
                    is.na(permit_amount$amount_entry)),])[1]==0){
 output7.1 <- rbind(output6.3, data.frame(step = "step 7.1", cat="entry permit
                                 pass related data missing",
                                 hh_unique_no = 99999,
                                 issue="No problems found",
                                 location_name_interviewer="Not applicable"))
 }else{
 permit_data_miss <- permit_amount[which(is.na(permit_amount$pass_permit_product)|
                                  is.na(permit_amount$number_collection)|
                                  is.na(permit_amount$amount_entry)),]
 permit_data_miss$step <- "Step 7.1"
 permit_data_miss$cat <- "entry permit pass related data missing"
 permit_data_miss$issue <- paste(ifelse(is.na(permit_amount$pass_permit_product),
                                "Product data missing",
                                ifelse(is.na(permit_amount$number_collection),
                                     "product collection number
                                     data missing",
                                     "product quantity collected
                                     data missing")))
 permit_data_miss <- merge(permit_data_miss, hh_info[,c("hh_unique_no",
                                       "location_name_interviewer")],
                    by.x = "household_hh_unique_no",
```

```
                    by.y = "hh_unique_no",
                    all.x=T)

 names(permit_data_miss)[1]<- "hh_unique_no"
 output7.1<- rbind(output6.3, permit_data_miss[,c("step", "cat",
                                  "hh_unique_no",
                                  "issue",
                                  "location_name_interviewer")])
 }



#-------------------------------------------------------------------------#
# 7.2 Tree and forest product for which pass permit is required is inconsistent
# Input: entry_fee_product_srf.csv; household.csv and
#       primary_tree_forest_products.csv
# Output: The list of hh for which the pass_permit_product will not match
#       with the list of primary product collection
#-------------------------------------------------------------------------#




library(ggplot2)
permit_amount$product_label <- ifelse(permit_amount$pass_permit_product==1,
                            "Nypa",
                            ifelse(permit_amount$pass_permit_product==2,
```

```
                    "Fish",
                    ifelse(permit_amount$pass_permit_product==3,
                        "Crabs",
                        ifelse(permit_amount$pass_permit_product==4,
                            "Honey",
                            ifelse(permit_amount$pass_permit_product==5,
                                "Beeswax",
                                ifelse(permit_amount$pass_permit_product==6,
                                    "Hantal",
                                    ifelse(permit_amount$pass_permit_product==7,
                                        "Hogla",
                                        ifelse(permit_amount$pass_permit_product==8,
                                            "Prawn",
                                            permit_amount$pass_permit_product_qualifier)))))))


ggplot(permit_amount, aes(x=product_label, y=amount_entry))+
  geom_boxplot()+
  xlab("Products")+ylab("Amount of pass permit per trip for each product (kg)")+
  ggtitle("Amount of different products taken pass permit")+
  theme(plot.title = element_text(hjust=0.5))




#-------------------------------------------------------------------------#
# 8 processed_tree_forest_product.csv
# 8.1 processed product produced but quanity not mentioned
#-------------------------------------------------------------------------#
```

```
pp <- pros_pordt


if(dim(pp[!is.na(pp$hh_pro_pdt)&is.na(pp$quantity_production),])[1]==0){
  output8.1 <- rbind(output7.1, data.frame(step = "step 8.1", cat="Prosess product quanitity not
mentioned",
                                           hh_unique_no = 99999,
                                           issue="No problems found",
                                           location_name_interviewer="Not applicable"))
}else{
  pp_data_miss <- pp[!is.na(pp$hh_pro_pdt)&is.na(pp$quantity_production),]
  pp_data_miss$step <- "Step 8.1"
  pp_data_miss$cat <- "Process product quanitity not mentioned"
  pp_data_miss$issue <- "Process product mentioned but quanitity not mentioned"
  pp_data_miss <- merge(pp_data_miss, hh_info[,c("hh_unique_no",
                                                 "location_name_interviewer")],
                        by.x = "household_hh_unique_no",
                        by.y = "hh_unique_no",
                        all.x=T)
  names(pp_data_miss)[1]<- "hh_unique_no"
  output8.1<- rbind(output7.1, pp_data_miss[,c("step", "cat",
                                               "hh_unique_no",
                                               "issue", "location_name_interviewer")])
}



#-----------------------------------------------------------------------------#
# 8.2 Don't hire labor for product processing but labor cost mentioned
#-----------------------------------------------------------------------------#
pp <- pros_pordt
```

```
if(dim(pp[pp$hire_labor==0 & (!is.na(pp$labor_cost)&pp$labor_cost>1),])[1]==0){
  output8.2 <- rbind(output8.1, data.frame(step = "step 8.2", cat="Processed product labor info inconsistent",
                                           hh_unique_no = 99999,
                                           issue="No problems found",
                                           location_name_interviewer="Not applicable"))
}else{
  pp_labor_inconsistency <- pp[pp$hire_labor==0 & (!is.na(pp$labor_cost)&pp$labor_cost>1),]
  pp_labor_inconsistency$step <- "Step 8.2"
  pp_labor_inconsistency$cat <- "Processed product labor info inconsistent"
  pp_labor_inconsistency$issue <- "Labor not hired but labor cost mentioned"
  pp_labor_inconsistency <- merge(pp_labor_inconsistency, hh_info[,c("hh_unique_no",
                                                                     "location_name_interviewer")],
                                  by.x = "household_hh_unique_no",
                                  by.y = "hh_unique_no",
                                  all.x=T)
  names(pp_labor_inconsistency)[1]<- "hh_unique_no"
  output8.2<- rbind(output8.1, pp_labor_inconsistency[,c("step", "cat",
                                                         "hh_unique_no",
                                                         "issue", "location_name_interviewer")])
}
```

```
#-----------------------------------------------------------------------#
# 8.3 Monthly Labor cost too low for processed product
#-----------------------------------------------------------------------#
```

```
pp <- pros_pordt


if(dim(pp[!is.na(pp$labor_cost) & pp$labor_cost<250,])[1]==0){
  output8.3 <- rbind(output8.2, data.frame(step = "step 8.3", cat="Monthly labor cost for processed product",
                                           hh_unique_no = 99999,
                                           issue="No problems found",
                                           location_name_interviewer="Not applicable"))
}else{
  pp_low_labor_cost <- pp[!is.na(pp$labor_cost) & pp$labor_cost<250,]
  pp_low_labor_cost$step <- "Step 8.3"
  pp_low_labor_cost$cat <- "Monthly labor cost for processed product"
  pp_low_labor_cost$issue <- "Monthly labor cost for processed product too low"
  pp_low_labor_cost <- merge(pp_low_labor_cost, hh_info[,c("hh_unique_no",
                                    "location_name_interviewer")],
                  by.x = "household_hh_unique_no",
                  by.y = "hh_unique_no",
                  all.x=T)
  names(pp_low_labor_cost)[1]<- "hh_unique_no"
  output8.3<- rbind(output8.2, pp_low_labor_cost[,c("step", "cat",
                                    "hh_unique_no",
                                    "issue", "location_name_interviewer")])
}




# 8.4 collection of timber as primary product for timber based processed product

pri_pros_prdt <- merge(hh_info[,c("hh_unique_no", "location_zone")],
for_prodt[for_prodt$forest_products==1,][,c("household_hh_unique_no", "quantity_collection")],
by.x="hh_unique_no", by.y = "household_hh_unique_no", all.x=T)
```

```
names(pri_pros_prdt)[3] <- "timber_collection_cft"

pri_pros_prdt <- merge(pri_pros_prdt,
for_prodt[for_prodt$forest_products==2,][,c("household_hh_unique_no", "quantity_collection")],
by.x="hh_unique_no", by.y = "household_hh_unique_no", all.x=T)

names(pri_pros_prdt)[4] <- "pole_collection_no"

pri_pros_prdt <- merge(pri_pros_prdt,
pros_pordt[pros_pordt$hh_pro_pdt==2,][,c("household_hh_unique_no", "quantity_production")],

        by.x="hh_unique_no", by.y = "household_hh_unique_no", all.x=T)

names(pri_pros_prdt)[5] <- "wood_furniture_prodt_no"

pri_pros_prdt <- merge(pri_pros_prdt,
pros_pordt[pros_pordt$hh_pro_pdt==3,][,c("household_hh_unique_no", "quantity_production")],

        by.x="hh_unique_no", by.y = "household_hh_unique_no", all.x=T)

names(pri_pros_prdt)[6] <- "wood_agri_appliance_prodt_no"

pri_pros_prdt <- merge(pri_pros_prdt,
for_prodt[for_prodt$forest_products==4,][,c("household_hh_unique_no", "quantity_collection")],
by.x="hh_unique_no", by.y = "household_hh_unique_no", all.x=T)

names(pri_pros_prdt)[7] <- "bamboo_collection_no"

pri_pros_prdt <- merge(pri_pros_prdt,
pros_pordt[pros_pordt$hh_pro_pdt==4,][,c("household_hh_unique_no", "quantity_production")],

        by.x="hh_unique_no", by.y = "household_hh_unique_no", all.x=T)

names(pri_pros_prdt)[8] <- "handicrafts_prodt_no"

write.csv(pri_pros_prdt, file="./SE-QAQC/Timber-Bamboo collection and furniture-Handicraft
production.csv", row.names = F)


# 8.5 data missing in processed_tree_forest_product

# 8.6.1 months_processed data missing

# 8.6.2 days_processed data missing

# 8.6.3 hours_processed data missing

# 8.6.4 raw material buy data missing

# 8.6.5 raw material cost data missing

# 8.6.6 labor hire data missing

# 8.6.7 labor cost data missing
```

```
# 8.6.8 processed product selling data missing
# 8.7 Labor cost inconsistent
# 8.8 Processed product no sold but quantity sold, hh_processed_selling,
#     months_selling, days_selling, hours_selling, income_processed
#     cost_selling_processed, transport_other_cost data mentioned
# 8.6 Production cost is higher than the selling const of processed product
# 8.7 Raw material bought but buying cost not mentioned
```

```
#-------------------------------------------------------------------------#
# 9.1 Cyclone
# Cat: HH damage under 96 category due to Cyclone
# Input: cyclone (damage_cyclone.csv)
# Output: list of hh_id with damage names under 96 category that need revision
#-------------------------------------------------------------------------#
cyclone_96 <- cyclone[cyclone$hh_damaged==96,]
unique(cyclone_96$hh_damaged_qualifier)
#review of the damage indicate that the names i.e. "Pady", "paddy" are
# not consistent

if(dim(cyclone_96)[1]==0){
```

```
  output9.1 <- rbind(output8.3, data.frame(step = "step 9.1", cat="Cyclone damage names",
                                           hh_unique_no = 99999,
                                           issue="No problems found",
                                           location_name_interviewer="Not applicable"))
}else{
  cyclone_96$step <- "Step 9.1"
  cyclone_96$cat <- "Cyclone damage names"
  cyclone_96$issue <- "damange names need rivision"
  cyclone_96 <- merge(cyclone_96, hh_info[,c("hh_unique_no",
                                             "location_name_interviewer")],
                      by.x = "household_hh_unique_no",
                      by.y = "hh_unique_no",
                      all.x=T)
  names(cyclone_96)[1] <- "hh_unique_no"
  output9.1 <- rbind(output8.3, cyclone_96[,c("step", "cat",
                                              "hh_unique_no",
                                              "issue",
                                              "location_name_interviewer")])
}



#-----------------------------------------------------------------------------#
# 10.1 Gender and age of HH members involved in processing the tree and forest products
# Cat: cells having unnecessary zero
# Input: gen_age_pdt.csv
# Output: list of cells having zero values uncecessarily [need to make vacant]
#-----------------------------------------------------------------------------#
library(dplyr)
```

```
zc <- filter(gen_pdt, male == 0 | female==0|third_gender==0)


if(dim(filter(gen_pdt, male == 0 | female==0|third_gender==0))[1]==0){
  output10.1 <- rbind(output9.1, data.frame(step = "step 10.1", cat="unnecessary zero in cells of
age_gen_pdt",
                                hh_unique_no = 99999,
                                issue="No problems found",
                                location_name_interviewer="Not applicable"))
  }else{
  zc <- filter(gen_pdt, male == 0 | female==0|third_gender==0)
  zc$step <- "Step 10.1"
  zc$cat <- "unnecessary zero in cells of age_gen_pdt"
  zc$issue <- "unnecessary zero in male, female or third gender column"
  zc <- merge(zc, hh_info[,c("hh_unique_no",
                      "location_name_interviewer")],
          by.x = "household_hh_unique_no",
          by.y = "hh_unique_no",
          all.x=T)
  names(zc)[1]<- "hh_unique_no"
  output10.1<- rbind(output9.1, zc[,c("step", "cat",
                                "hh_unique_no",
                                "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------------#
# 10.2 finding duplicated rows in the data frame upto age class
# Cat: duplicated rows in gen_age_pdt
# Input: gen_age_pdt.csv
```

```
# Output: list of duplicated rows upt to age class
#------------------------------------------------------------------------#
#finding the duplicated rows for hh_id and age class
gen_pdt[duplicated(gen_pdt[,c("household_hh_unique_no", "age_class")]),]


if(dim(gen_pdt[duplicated(gen_pdt[,c("household_hh_unique_no", "age_class")]),])[1]==0){
  output10.2 <- rbind(output10.1, data.frame(step = "step 10.2", cat="duplicated rows in gen_age_pdt",
                                  hh_unique_no = 99999,
                                  issue="No problems found",
                                  location_name_interviewer="Not applicable"))
}else{
  dr <- gen_pdt[duplicated(gen_pdt[,c("household_hh_unique_no", "age_class")]),] #dr = duplicated
rows
  dr$step <- "Step 10.2"
  dr$cat <- "duplicated rows in gen_age_pdt"
  dr$issue <- "row duplicated up to age class"
  dr <- merge(dr, hh_info[,c("hh_unique_no",
                        "location_name_interviewer")],
            by.x = "household_hh_unique_no",
            by.y = "hh_unique_no",
            all.x=T)
  names(dr)[1]<- "hh_unique_no"
  output10.2<- rbind(output10.1, dr[,c("step", "cat",
                                "hh_unique_no",
                                "issue", "location_name_interviewer")])
}


#------------------------------------------------------------------------#
```

```
# 10.3 finding mismatched hh_id in gen_age_pdt with processed_tree_forest_product

# Cat: missmatch hh_id in gen_age_pdt

# Input: gen_age_pdt.csv and processed_tree_forest_product.csv

# Output: list of mismatched hh_id

#-------------------------------------------------------------------------#

#hh_id not in processed_tree_forest_product.csv

test1<- gen_pdt$household_hh_unique_no[!(gen_pdt$household_hh_unique_no %in%
pros_pordt$household_hh_unique_no)]

#hh_id not in gen_age_pdt.csv

test2 <- pros_pordt$household_hh_unique_no[!(pros_pordt$household_hh_unique_no %in%
gen_pdt$household_hh_unique_no)]

test <- union(test1, test2)



if(dim(gen_pdt[gen_pdt$household_hh_unique_no %in% test,])[1]==0){

  output10.3 <- rbind(output10.2, data.frame(step = "step 10.3", cat="mismatch betwn gen_age_pdt and
processed_product",

                                             hh_unique_no = 99999,

                                             issue="No problems found",

                                             location_name_interviewer="Not applicable"))

}else{

  mismatch <- gen_pdt[gen_pdt$household_hh_unique_no %in% test,] #mismatch = mismatching rows
between gen_pdt and processed_tree_forest_product

  mismatch$step <- "Step 10.3"

  mismatch$cat <- "mismatch betwn gen_age_pdt and processed_product"

  mismatch$issue <- ifelse(mismatch$household_hh_unique_no==test1, "hh_id missing in
processed_tree_forest_product",

                           "hh_id missing in gen_pdt")

  mismatch <- merge(mismatch, hh_info[,c("hh_unique_no",

                                         "location_name_interviewer")],
```

```
          by.x = "household_hh_unique_no",
          by.y = "hh_unique_no",
          all.x=T)
  names(mismatch)[1]<- "hh_unique_no"
  output10.3<- rbind(output10.2, mismatch[,c("step", "cat",
                                "hh_unique_no",
                                "issue", "location_name_interviewer")])
}



#---------------------------------------------------------------------------#
# 10.4 number of hh members in gen_age_pdt is greater than the hh members in household.csv
# Cat: inconsistency in hh members of gen_age_pdt
# Input: gen_age_pdt.csv and household.csv
# Output: list of hh_id having greater hh members in gen_age_pdt than household.csv
#---------------------------------------------------------------------------#
test10.4 <- gen_pdt
test10.4[,4:6][is.na(test10.4[,4:6])] <- 0
test10.4$tot_hh_mem <- test10.4$male+test10.4$female+test10.4$third_gender
test10.4 <- aggregate(data=test10.4, tot_hh_mem~household_hh_unique_no, FUN=sum)
test10.4 <- merge(test10.4, hh_info[,c("hh_unique_no", "demographic_info_hh_members")],
                          by.x = "household_hh_unique_no",
                          by.y="hh_unique_no", all.x=T)
test10.4 <- test10.4[test10.4$tot_hh_mem > test10.4$demographic_info_hh_members,]

if(dim(test10.4[test10.4$tot_hh_mem > test10.4$demographic_info_hh_members,])[1]==0){
  output10.4 <- rbind(output10.3, data.frame(step = "step 10.4", cat="inconsistency in hh members of
gen_age_pdt",
                                hh_unique_no = 99999,
```

```
                                issue="No problems found",
                                location_name_interviewer="Not applicable"))
}else{
 test10.4 <- test10.4[test10.4$tot_hh_mem > test10.4$demographic_info_hh_members,]
 test10.4$step <- "Step 10.4"
 test10.4$cat <- "inconsistency in hh members of gen_age_pdt"
 test10.4$issue <- "hh members in gen_age_pdt > hh members in household"
 test10.4 <- merge(test10.4, hh_info[,c("hh_unique_no",
                                "location_name_interviewer")],
           by.x = "household_hh_unique_no",
           by.y = "hh_unique_no",
           all.x=T)
 names(test10.4)[1]<- "hh_unique_no"
 output10.4<- rbind(output10.3, test10.4[,c("step", "cat",
                                "hh_unique_no",
                                "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------#
# 11.1 Gender and age of HH members involved in selling the tree and forest products
# Cat: cells having unnecessary zero
# Input: gen_age_cell.csv
# Output: list of cells having zero values uncecessarily [need to make vacant]
#-------------------------------------------------------------------------#
zc <- filter(gen_sell, male == 0 | female==0|third_gender==0) #zc = zero cells

if(dim(filter(gen_sell, male == 0 | female==0|third_gender==0))[1]==0){
```

```
 output11.1 <- rbind(output10.4, data.frame(step = "step 11.1", cat="unnecessary zero in cells of
age_gen_sell",
                                hh_unique_no = 99999,
                                issue="No problems found",
                                location_name_interviewer="Not applicable"))
}else{
 zc <- filter(gen_sell, male == 0 | female==0|third_gender==0)
 zc$step <- "Step 11.1"
 zc$cat <- "unnecessary zero in cells of age_gen_sell"
 zc$issue <- "unnecessary zero in male, female or third gender column"
 zc <- merge(zc, hh_info[,c("hh_unique_no",
                     "location_name_interviewer")],
          by.x = "household_hh_unique_no",
          by.y = "hh_unique_no",
          all.x=T)
 names(zc)[1]<- "hh_unique_no"
 output11.1<- rbind(output10.4, zc[,c("step", "cat",
                              "hh_unique_no",
                              "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------------#
# 11.2 finding duplicated rows in the data frame upto age class
# Cat: duplicated rows in gen_age_sell
# Input: gen_age_sell.csv
# Output: list of duplicated rows upto age class
#-------------------------------------------------------------------------------#
#finding the duplicated rows for hh_id and age class
```

```
gen_sell[duplicated(gen_sell[,c("household_hh_unique_no", "age_class")]),]


if(dim(gen_sell[duplicated(gen_sell[,c("household_hh_unique_no", "age_class")]),])[1]==0){
  output11.2 <- rbind(output11.1, data.frame(step = "step 11.2", cat="duplicated rows in gen_age_sell",
                                             hh_unique_no = 99999,
                                             issue="No problems found",
                                             location_name_interviewer="Not applicable"))
}else{
  dr <- gen_sell[duplicated(gen_sell[,c("household_hh_unique_no", "age_class")]),] #dr = duplicated
rows
  dr$step <- "Step 11.2"
  dr$cat <- "duplicated rows in gen_age_pdt"
  dr$issue <- "row duplicated up to age class"
  dr <- merge(dr, hh_info[,c("hh_unique_no",
                             "location_name_interviewer")],
              by.x = "household_hh_unique_no",
              by.y = "hh_unique_no",
              all.x=T)
  names(dr)[1]<- "hh_unique_no"
  output11.2<- rbind(output11.1, dr[,c("step", "cat",
                                       "hh_unique_no",
                                       "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------------#
# 11.3 finding mismatched hh_id in gen_age_pdt with processed_tree_forest_product
# Cat: missmatch hh_id in gen_age_sell
# Input: gen_age_sell.csv and processed_tree_forest_product.csv
```

```
# Output: list of mismatched hh_id

#-------------------------------------------------------------------------#

#hh_id not in sell list of processed_tree_forest_product.csv

test1<- gen_sell$household_hh_unique_no[!(gen_sell$household_hh_unique_no %in%
pros_pordt[pros_pordt$sell_processed==1,]$household_hh_unique_no)]

#hh_id not in gen_age_pdt.csv

test2 <-
pros_pordt[pros_pordt$sell_processed==1,]$household_hh_unique_no[!(pros_pordt[pros_pordt$sell_p
rocessed==1,]$household_hh_unique_no %in% gen_sell$household_hh_unique_no)]

test <- data.frame(hh_unique_no = union(test2, test1))



if(dim(test)[1]==0){

  output11.3 <- rbind(output11.2, data.frame(step = "step 11.3", cat="mismatch betwn gen_age_sell and
processed_product",

                                             hh_unique_no = 99999,

                                             issue="No problems found",

                                             location_name_interviewer="Not applicable"))

  }else{

  test$step <- "Step 11.3"

  test$cat <- "mismatch betwn gen_age_sell and processed_product"

  test$issue <- ifelse(test$hh_unique_no==test1, "hh_id missing in processed_tree_forest_product",

                       "hh_id missing in gen_age_sell")

  test <- merge(test, hh_info[,c("hh_unique_no",

                                 "location_name_interviewer")],

                by = "hh_unique_no", all.x=T)

  output11.3 <- rbind(output11.2, test[,c("step", "cat",

                                          "hh_unique_no",

                                          "issue", "location_name_interviewer")])

}
```

```
#-----------------------------------------------------------------------------#
# 11.4 number of hh members in gen_age_sell is greater than the hh members in household.csv
# Cat: inconsistency in hh members of gen_age_sell
# Input: gen_age_sell.csv and household.csv
# Output: list of hh_id having greater hh members in gen_age_sell than household.csv
#-----------------------------------------------------------------------------#
test11.4 <- gen_sell
test11.4[,4:6][is.na(test11.4[,4:6])] <- 0
test11.4$tot_hh_mem <- test11.4$male+test11.4$female+test11.4$third_gender
test11.4 <- aggregate(data=test11.4, tot_hh_mem~household_hh_unique_no, FUN=sum)
test11.4 <- merge(test11.4, hh_info[,c("hh_unique_no", "demographic_info_hh_members")],
          by.x = "household_hh_unique_no",
          by.y="hh_unique_no", all.x=T)
test11.4 <- test11.4[test11.4$tot_hh_mem > test11.4$demographic_info_hh_members,]

if(dim(test11.4[test11.4$tot_hh_mem > test11.4$demographic_info_hh_members,])[1]==0){
  output11.4 <- rbind(output11.3, data.frame(step = "step 11.4", cat="inconsistency in hh members of
gen_age_sell",
                                             hh_unique_no = 99999,
                                             issue="No problems found",
                                             location_name_interviewer="Not applicable"))
}else{
  test11.4 <- test11.4[test11.4$tot_hh_mem > test11.4$demographic_info_hh_members,]
  test11.4$step <- "Step 11.4"
  test11.4$cat <- "inconsistency in hh members of gen_age_pdt"
  test11.4$issue <- "hh members in gen_age_pdt > hh members in household"
  test11.4 <- merge(test11.4, hh_info[,c("hh_unique_no",
```

```
                        "location_name_interviewer")],
            by.x = "household_hh_unique_no",
            by.y = "hh_unique_no",
            all.x=T)
  names(test11.4)[1]<- "hh_unique_no"
  output11.4<- rbind(output11.3, test11.4[,c("step", "cat",
                        "hh_unique_no",
                        "issue", "location_name_interviewer")])
}



#-----------------------------------------------------------------------------#
# 12.1 hh energy dependence data missing
# Cat: missing data in hh_dependence
# Input: hh_dependence.csv
# Output: list of cells missing energy related data
#-----------------------------------------------------------------------------#
dm <- filter(dependence, quantity_ener_cons == 0 | is.na(quantity_ener_cons) | (money_energy==1 & is.na(expenditure_energy))) #dm = data missing


if(dim(filter(dependence, quantity_ener_cons == 0 | is.na(quantity_ener_cons) | (money_energy==1 & is.na(expenditure_energy))))[1]==0){
  output12.1 <- rbind(output11.4, data.frame(step = "step 12.1", cat="missing data in hh_dependence",
                        hh_unique_no = 99999,
                        issue="No problems found",
                        location_name_interviewer="Not applicable"))
}else{
  dm <- filter(dependence, quantity_ener_cons == 0 | is.na(quantity_ener_cons) | (money_energy==1 & is.na(expenditure_energy)))
  dm$step <- "Step 12.1"
```

```
  dm$cat <- "missing data in hh_dependence"
  dm$issue <- ifelse(dm$quantity_ener_cons==0 | is.na(dm$quantity_ener_cons), "energy consumption quantity data missing",
                     "energy expenditure data missing")
  dm <- merge(dm, hh_info[,c("hh_unique_no",
                             "location_name_interviewer")],
              by.x = "household_hh_unique_no",
              by.y = "hh_unique_no",
              all.x=T)
  names(dm)[1]<- "hh_unique_no"
  output12.1<- rbind(output11.4, dm[,c("step", "cat",
                                       "hh_unique_no",
                                       "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------------#
# 12.2 finding duplicated rows in the data frame upto age class
# Cat: duplicated rows in hh_dependence
# Input: hh_dependence.csv
# Output: list of duplicated rows upto money spent for energy
#-------------------------------------------------------------------------------#
#finding the duplicated rows for hh_id and age class
dependence[duplicated(dependence[,c("household_hh_unique_no", "type_energy_use",
"money_energy")]),]


if(dim(dependence[duplicated(dependence[,c("household_hh_unique_no", "type_energy_use",
"money_energy")]),])[1]==0){
  output12.2 <- rbind(output12.1, data.frame(step = "step 12.2", cat="duplicated rows in
hh_dependence",
```

```
                                hh_unique_no = 99999,

                                issue="No problems found",

                                location_name_interviewer="Not applicable"))

}else{

  dr <- dependence[duplicated(dependence[,c("household_hh_unique_no", "type_energy_use",
"money_energy")]),] #dr = duplicated rows

  dr$step <- "Step 12.2"

  dr$cat <- "duplicated rows in hh_dependence"

  dr$issue <- paste("hh position", dr$X_gen_age_pdt_position,"duplicated upto money spent for
energy")

  dr <- merge(dr, hh_info[,c("hh_unique_no",

                        "location_name_interviewer")],

          by.x = "household_hh_unique_no",

          by.y = "hh_unique_no",

          all.x=T)

  names(dr)[1]<- "hh_unique_no"

  output12.2<- rbind(output12.1, dr[,c("step", "cat",

                                "hh_unique_no",

                                "issue", "location_name_interviewer")])

}



#-------------------------------------------------------------------------#

# 12.3 finding outliers in quantity of energy consuption

# Cat: energy quantity and expenditure outliers in hh_dependence

# Input: hh_dependence.csv

# Output: list of hh_id having outliers in energy quanity and expenditure

#-------------------------------------------------------------------------#

test12.3 <- aggregate(data=dependence, cbind(quantity_ener_cons,expenditure_energy)
~household_hh_unique_no, FUN=sum)
```

```
if(dim(test12.3[test12.3$quantity_ener_cons>1000|test12.3$expenditure_energy>2500,])[1]==0){
  output12.3 <- rbind(output12.2, data.frame(step = "step 12.3", cat="energy quantity and expenditure outliers in hh_dependence",
                                             hh_unique_no = 99999,
                                             issue="No problems found",
                                             location_name_interviewer="Not applicable"))
}else{
  outlier <- test12.3[test12.3$quantity_ener_cons>1000|test12.3$expenditure_energy>2500,]
  outlier$step <- "Step 12.3"
  outlier$cat <- "energy quantity and expenditure outliers in hh_dependence"
  outlier$issue <- ifelse(outlier$quantity_ener_cons>1000,
                          paste("hh energy consumption too high",outlier$quantity_ener_cons,"kg"),
                          paste("hh energy expenditure too high",outlier$expenditure_energy,"TK"))
  outlier <- merge(outlier, hh_info[,c("hh_unique_no",
                                       "location_name_interviewer")],
                   by.x = "household_hh_unique_no", by.y = "hh_unique_no", all.x=T)
  names(outlier)[1] <- "hh_unique_no"
  output12.3 <- rbind(output12.2, outlier[,c("step", "cat",
                                             "hh_unique_no",
                                             "issue", "location_name_interviewer")])
}


#-----------------------------------------------------------------------------#
# 12.4 energy type LP gas and kerosene but no expenditure data provided
# Cat: energy expenditure data absent for LP gas and kerosene
# Input: hh_dependence.csv
# Output: list of hh_id using LP gas and kerosene but missing energy expenditure data
#-----------------------------------------------------------------------------#
```

```
test12.4 <- filter(dependence, money_energy==0 & (type_energy_use==5 | type_energy_use==6))


if(dim(test12.4)[1]==0){
  output12.4 <- rbind(output12.3, data.frame(step = "step 12.4", cat="energy expenditure data absent for LP gas and kerosene",
                                             hh_unique_no = 99999,
                                             issue="No problems found",
                                             location_name_interviewer="Not applicable"))
}else{
  test12.4$step <- "Step 12.4"
  test12.4$cat <- "energy expenditure data absent for LP gas and kerosene"
  test12.4$issue <- ifelse(test12.4$type_energy_use==5,
                           "energy expenditure data absent for LP gas",
                           "energy expenditure data absent for Kerosene")
  test12.4 <- merge(test12.4, hh_info[,c("hh_unique_no",
                                         "location_name_interviewer")],
                    by.x = "household_hh_unique_no",
                    by.y = "hh_unique_no",
                    all.x=T)
  names(test12.4)[1]<- "hh_unique_no"
  output12.4<- rbind(output12.3, test12.4[,c("step", "cat",
                                             "hh_unique_no",
                                             "issue", "location_name_interviewer")])
}



#-----------------------------------------------------------------------------#
# 12.5 unnecessary data in woodfuel dependency column of hh_dependence
# Cat: unnecessary data on woodfuel dependency level
```

```
# Input: hh_dependence.csv

# Output: list of hh_id having unnecessary data in woodfuel dependency column of hh_dependence.csv

#-----------------------------------------------------------------------------#

test12.5 <- filter(dependence, (type_energy_use!=1 & type_energy_use!=2) &
!is.na(dependency_woodfuel))


if(dim(test12.5)[1]==0){

  output12.5 <- rbind(output12.4, data.frame(step = "step 12.5", cat="unnecessary data on woodfuel
dependency level",

                                             hh_unique_no = 99999,

                                             issue="No problems found",

                                             location_name_interviewer="Not applicable"))

}else{

  test12.5 <- filter(dependence, (type_energy_use!=1 & type_energy_use!=2) &
!is.na(dependency_woodfuel))

  test12.5$step <- "Step 12.5"

  test12.5$cat <- "unnecessary data on woodfuel dependency level"

  test12.5$issue <- paste("fuel type", test12.5$type_energy_use, "but woodfuel dependency level
mentioned")

  test12.5 <- merge(test12.5, hh_info[,c("hh_unique_no",

                                         "location_name_interviewer")],

                    by.x = "household_hh_unique_no",

                    by.y = "hh_unique_no",

                    all.x=T)

  names(test12.5)[1]<- "hh_unique_no"

  output12.5<- rbind(output12.4, test12.5[,c("step", "cat",

                                             "hh_unique_no",

                                             "issue", "location_name_interviewer")])

}
```

```
#------------------------------------------------------------------------#
# 13.1 data missing in hh_disturbance
# Cat: missing data in hh_disturbance
# Input: hh_disturbance.csv
# Output: list of cells disturbance data
#------------------------------------------------------------------------#
dm <- filter(distrb, is.na(X_hh_disturbance_position) | X_hh_disturbance_position==0 |
is.na(type_disturbances)) #dm = data missing


if(dim(filter(distrb, is.na(X_hh_disturbance_position) | X_hh_disturbance_position==0 |
is.na(type_disturbances)))[1]==0){
  output13.1 <- rbind(output12.5, data.frame(step = "step 13.1", cat="missing data in hh_disturbance",
                                             hh_unique_no = 99999,
                                             issue="No problems found",
                                             location_name_interviewer="Not applicable"))
}else{
  dm <- filter(distrb, is.na(X_hh_disturbance_position) | X_hh_disturbance_position==0 |
is.na(type_disturbances))
  dm$step <- "Step 13.1"
  dm$cat <- "missing data in hh_disturbance"
  dm$issue <- ifelse(is.na(dm$X_hh_disturbance_position) | dm$X_hh_disturbance_position==0,
  "data missing in hh_disturbance_position", "data missing in type_disturbances")
  dm <- merge(dm, hh_info[,c("hh_unique_no",
                             "location_name_interviewer")],
              by.x = "household_hh_unique_no",
              by.y = "hh_unique_no",
              all.x=T)
  names(dm)[1]<- "hh_unique_no"
  output13.1<- rbind(output12.5, dm[,c("step", "cat",
```

```
                        "hh_unique_no",
                        "issue", "location_name_interviewer")])
}

#-----------------------------------------------------------------------------#
# 13.2 finding duplicated rows in the data frame upto disturbance types
# Cat: duplicated rows in hh_disturbance
# Input: hh_disturbance.csv
# Output: list of duplicated rows upto disturbance types
#-----------------------------------------------------------------------------#
#finding the duplicated rows for hh_id and disturbance types
distrb[duplicated(distrb[,c("household_hh_unique_no", "type_disturbances")]),]

if(dim(distrb[duplicated(distrb[,c("household_hh_unique_no", "type_disturbances")]),])[1]==0){
  output13.2 <- rbind(output13.1, data.frame(step = "step 13.2", cat="duplicated rows in
hh_disturbance",
                                hh_unique_no = 99999,
                                issue="No problems found",
                                location_name_interviewer="Not applicable"))
}else{
  dr <- distrb[duplicated(distrb[,c("household_hh_unique_no", "type_disturbances")]),] #dr = duplicated
rows
  dr$step <- "Step 13.2"
  dr$cat <- "duplicated rows in hh_disturbance"
  dr$issue <- paste("hh position", dr$X_hh_disturbance_position,"duplicated upto disturbance types")
  dr <- merge(dr, hh_info[,c("hh_unique_no",
                      "location_name_interviewer")],
          by.x = "household_hh_unique_no",
          by.y = "hh_unique_no",
```

```
            all.x=T)
  names(dr)[1]<- "hh_unique_no"
  output13.2<- rbind(output13.1, dr[,c("step", "cat",
                                       "hh_unique_no",
                                       "issue", "location_name_interviewer")])
}




#-----------------------------------------------------------------------------#
# 14.1 data missing in human_mande_disturbance
# Cat: missing data in human_made_disturbances
# Input: human_made_disturbances.csv
# Output: list of cells disturbance data
#-----------------------------------------------------------------------------#
dm <- filter(hum_distrb, is.na(disturbances_human) | is.na(severity_dist_human) |
is.na(location_human_dis.1.)) #dm = data missing


if(dim(filter(hum_distrb, is.na(disturbances_human) | is.na(severity_dist_human) |
is.na(location_human_dis.1.)))[1]==0){
  output14.1 <- rbind(output13.2, data.frame(step = "step 14.1", cat="missing data in
human_made_disturbances",
                                     hh_unique_no = 99999,
                                     issue="No problems found",
                                     location_name_interviewer="Not applicable"))
}else{
  dm <- filter(hum_distrb, is.na(disturbances_human) | is.na(severity_dist_human) |
is.na(location_human_dis.1.))
  dm$step <- "Step 14.1"
  dm$cat <- "missing data in human_made_disturbances"
```

```
 dm$issue <- ifelse(is.na(dm$disturbances_human),
                    "data missing in disturbances_human",
                    ifelse(is.na(dm$severity_dist_human),
                           "data missing in severity_dist_human",
                           "data missing in location_human_dis.1."))
 dm <- merge(dm, hh_info[,c("hh_unique_no",
                            "location_name_interviewer")],
             by.x = "household_hh_unique_no",
             by.y = "hh_unique_no",
             all.x=T)
 names(dm)[1]<- "hh_unique_no"
 output14.1<- rbind(output13.2, dm[,c("step", "cat",
                                      "hh_unique_no",
                                      "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------#
# 14.2 finding duplicated rows in the data frame upto disturbances_human
# Cat: duplicated rows in human_made_disturbance
# Input: human_made_disturbance.csv
# Output: list of duplicated rows upto disturbances_human
#-------------------------------------------------------------------------#
#finding the duplicated rows for hh_id and age class
hum_distrb[hum_distrb$disturbances_human!=96,][duplicated(hum_distrb[hum_distrb$disturbances_human!=96,][,c("household_hh_unique_no", "disturbances_human")]),]


if(dim(hum_distrb[hum_distrb$disturbances_human!=96,][duplicated(hum_distrb[hum_distrb$disturbances_human!=96,][,c("household_hh_unique_no", "disturbances_human")]),])[1]==0){
```

```
  output14.2 <- rbind(output14.1, data.frame(step = "step 14.2", cat="duplicated rows in
human_made_disturbance",
                                  hh_unique_no = 99999,
                                  issue="No problems found",
                                  location_name_interviewer="Not applicable"))
}else{
  dr <-
hum_distrb[hum_distrb$disturbances_human!=96,][duplicated(hum_distrb[hum_distrb$disturbances_h
uman!=96,][,c("household_hh_unique_no", "disturbances_human")]),] #dr = duplicated rows
  dr$step <- "Step 14.2"
  dr$cat <- "duplicated rows in human_made_disturbance"
  dr$issue <- paste("hh position", dr$X_human_made_disturbances_position,"duplicated upto
disturbances_human")
  dr <- merge(dr, hh_info[,c("hh_unique_no",
                        "location_name_interviewer")],
          by.x = "household_hh_unique_no",
          by.y = "hh_unique_no",
          all.x=T)
  names(dr)[1]<- "hh_unique_no"
  output14.2<- rbind(output14.1, dr[,c("step", "cat",
                                "hh_unique_no",
                                "issue", "location_name_interviewer")])
}
#-----------------------------------------------------------------------------#
# 14.3 finding mismatched hh_id in human_made_disturbance with hh_disturbance
# Cat: missmatch of hh_id between hh_disturbance and human_made_disturbance
# Input: human_made_disturbances.csv and hh_disturbance.csv
# Output: list of mismatched hh_id
```

```
#-----------------------------------------------------------------------------#

#hh_id not in hh_disturbance.csv

test1<- hum_distrb$household_hh_unique_no[!(hum_distrb$household_hh_unique_no %in%
distrb[distrb$type_disturbances==1,]$household_hh_unique_no)]

#hh_id not human_made_disturbance.csv

test2 <-
distrb[distrb$type_disturbances==1,]$household_hh_unique_no[!(distrb[distrb$type_disturbances==1,]
$household_hh_unique_no %in% hum_distrb$household_hh_unique_no)]

test <- data.frame(hh_unique_no = union(test2, test1))




if(dim(test)[1]==0){

  output14.3 <- rbind(output14.2, data.frame(step = "step 14.3", cat="missmatch of hh_id btwn
hh_disturbance and human_made_disturbance",

                                             hh_unique_no = 99999,

                                             issue="No problems found",

                                             location_name_interviewer="Not applicable"))

}else{

  test$step <- "Step 14.3"

  test$cat <- "missmatch of hh_id btwn hh_disturbance and human_made_disturbance"

  test$issue <- ifelse(test$hh_unique_no==test1, "hh_id missing in hh_disturbance",

                       "hh_id missing in human_made_disturbance")

  mismatch <- merge(test, hh_info[,c("hh_unique_no",

                                     "location_name_interviewer")],

                    by = "hh_unique_no", all.x=T)

  output14.3 <- rbind(output14.2, mismatch[,c("step", "cat",

                                              "hh_unique_no",

                                              "issue", "location_name_interviewer")])

}
```

```
#-----------------------------------------------------------------------------#

# 14.4 non-uniform spelling under 96 disturbance category of human_made_disturbance

# Cat: non-uniform spelling under 96 disturbance category

# Input: human_made_disturbances.csv

# Output: list of non-uniform spelling hh_id

#-----------------------------------------------------------------------------#

distb_96 <-
sort(unique(hum_distrb[hum_distrb$disturbances_human==96,]$disturbances_human_qualifier))

distb_96 <- paste(as.character(distb_96), collapse="; ")


output14.4 <- rbind(output14.3, data.frame(step = "step 14.4", cat="non-uniform spelling under 96
human made disturbance category",

                                           hh_unique_no = 99999,

                                           issue=paste("These (", distb_96, ") need to be of uniform spelling", sep = ""),

                                           location_name_interviewer="Not applicable"))


#-----------------------------------------------------------------------------#

# 15.1 data missing in natural_disturbance

# Cat: missing data in natural_disturbance

# Input: natural_disturbance.csv

# Output: list of cells disturbance data

#-----------------------------------------------------------------------------#

dm <- filter(nat_distrb, is.na(natural_disturbances) | is.na(severity_dist_natural) |
is.na(location_natural_dis.1.)) #dm = data missing


if(dim(filter(nat_distrb, is.na(natural_disturbances) | is.na(severity_dist_natural) |
is.na(location_natural_dis.1.)))[1]==0){

  output15.1 <- rbind(output14.4, data.frame(step = "step 15.1", cat="missing data in
natural_disturbances",
```

```
                                hh_unique_no = 99999,
                                issue="No problems found",
                                location_name_interviewer="Not applicable"))
}else{
  dm <- filter(nat_distrb, is.na(natural_disturbances) | is.na(severity_dist_natural) |
is.na(location_natural_dis.1.))
  dm$step <- "Step 15.1"
  dm$cat <- "missing data in natural_disturbances"
  dm$issue <- ifelse(is.na(dm$natural_disturbances),
                  "data missing in natural_disturbances",
                  ifelse(is.na(dm$severity_dist_natural),
                      "data missing in severity_dist_natural",
                      "data missing in location_natural_dis.1."))
  dm <- merge(dm, hh_info[,c("hh_unique_no",
                       "location_name_interviewer")],
            by.x = "household_hh_unique_no",
            by.y = "hh_unique_no",
            all.x=T)
  names(dm)[1]<- "hh_unique_no"
  output15.1<- rbind(output14.4, dm[,c("step", "cat",
                                "hh_unique_no",
                                "issue", "location_name_interviewer")])
}



#-----------------------------------------------------------------------------#
# 15.2 finding duplicated rows in the data frame upto natural_disturbances
# Cat: duplicated rows in natural_disturbance
# Input: natural_disturbance.csv
```

```
# Output: list of duplicated rows upto natural_disturbances

#-----------------------------------------------------------------------------#

#finding the duplicated rows for hh_id and age class

nat_distrb[nat_distrb$natural_disturbances!=96,][duplicated(nat_distrb[nat_distrb$natural_disturbances!=96,][,c("household_hh_unique_no", "natural_disturbances")]),]


if(dim(nat_distrb[nat_distrb$natural_disturbances!=96,][duplicated(nat_distrb[nat_distrb$natural_disturbances!=96,][,c("household_hh_unique_no", "natural_disturbances")]),])[1]==0){

  output15.2 <- rbind(output15.1, data.frame(step = "step 15.2", cat="duplicated rows in natural_disturbance",

                                             hh_unique_no = 99999,

                                             issue="No problems found",

                                             location_name_interviewer="Not applicable"))

}else{

  dr <- nat_distrb[nat_distrb$natural_disturbances!=96,][duplicated(nat_distrb[nat_distrb$natural_disturbances!=96,][,c("household_hh_unique_no", "natural_disturbances")]),] #dr = duplicated rows

  dr$step <- "Step 15.2"

  dr$cat <- "duplicated rows in natural_disturbance"

  dr$issue <- paste("hh position", dr$X_natural_disturbance_position,"duplicated upto natural_disturbances")

  dr <- merge(dr, hh_info[,c("hh_unique_no",

                             "location_name_interviewer")],

              by.x = "household_hh_unique_no",

              by.y = "hh_unique_no",

              all.x=T)

  names(dr)[1]<- "hh_unique_no"

  output15.2<- rbind(output15.1, dr[,c("step", "cat",

                                       "hh_unique_no",

                                       "issue", "location_name_interviewer")])

}
```

```
#------------------------------------------------------------------------------#

# 15.3 finding mismatched hh_id in natural_disturbance with hh_disturbance

# Cat: missmatch of hh_id between hh_disturbance and natural_disturbance

# Input: natrual_disturbance.csv and hh_disturbance.csv

# Output: list of mismatched hh_id

#------------------------------------------------------------------------------#

#hh_id not in hh_disturbance.csv

test1<- nat_distrb$household_hh_unique_no[!(nat_distrb$household_hh_unique_no %in%
distrb[distrb$type_disturbances==2,]$household_hh_unique_no)]

#hh_id not natural_disturbance.csv

test2 <-
distrb[distrb$type_disturbances==2,]$household_hh_unique_no[!(distrb[distrb$type_disturbances==2,]
$household_hh_unique_no %in% nat_distrb$household_hh_unique_no)]

test <- data.frame(hh_unique_no = union(test2, test1))



if(dim(test)[1]==0){

  output15.3 <- rbind(output15.2, data.frame(step = "step 15.3", cat="missmatch of hh_id between
hh_disturbance and natural_disturbance",

                                             hh_unique_no = 99999,

                                             issue="No problems found",

                                             location_name_interviewer="Not applicable"))

}else{

  test$step <- "Step 15.3"

  test$cat <- "missmatch of hh_id between hh_disturbance and natural_disturbance"

  test$issue <- "hh_id mismatch between hh_disturbance and natrual_disturbance"

  mismatch <- merge(test, hh_info[,c("hh_unique_no",

                                     "location_name_interviewer")],
```

```
                by = "hh_unique_no", all.x=T)
  output15.3 <- rbind(output15.2, mismatch[,c("step", "cat",
                                    "hh_unique_no",
                                    "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------#
# 15.4 non-uniform spelling under 96 disturbance category of natural_disturbance
# Cat: non-uniform spelling under 96 disturbance category
# Input: natural
# Output: list of non-uniform spelling hh_id
#-------------------------------------------------------------------------#
distb_96 <-
sort(unique(nat_distrb[nat_distrb$natural_disturbances==96,]$natural_disturbances_qualifier))
distb_96 <- paste(as.character(distb_96), collapse="; ")


output15.4 <- rbind(output15.3, data.frame(step = "step 15.4", cat="non-uniform spelling under 96
natural disturbance category",
                                    hh_unique_no = 99999,
                                    issue=paste("These (", distb_96, ") need to be of uniform spelling", sep = ""),
                                    location_name_interviewer="Not applicable"))



#-------------------------------------------------------------------------#
# 16.1 Gender and age of HH members involved in collection of primary tree and forest products
# Cat: cells having unnecessary zero
# Input: hh_gender_age.csv
# Output: list of cells having uncecessary zero [need to make vacant]
```

```
#-----------------------------------------------------------------------------#

zc <- filter(gen_age, male == 0 | female==0|third_gender==0) #zc = zero cells; gen_age=
hh_gender_age.csv


if(dim(filter(gen_age, male == 0 | female==0|third_gender==0))[1]==0){

  output16.1 <- rbind(output15.4, data.frame(step = "step 16.1", cat="unnecessary zero in
hh_gender_age",

                                  hh_unique_no = 99999,

                                  issue="No problems found",

                                  location_name_interviewer="Not applicable"))

}else{

  zc <- filter(gen_age, male == 0 | female==0|third_gender==0)

  zc$step <- "Step 16.1"

  zc$cat <- "unnecessary zero in hh_gender_age"

  zc$issue <- "unnecessary zero in male, female or third gender column"

  zc <- merge(zc, hh_info[,c("hh_unique_no",

                        "location_name_interviewer")],

          by.x = "household_hh_unique_no",

          by.y = "hh_unique_no",

          all.x=T)

  names(zc)[1]<- "hh_unique_no"

  output16.1<- rbind(output15.4, zc[,c("step", "cat",

                                "hh_unique_no",

                                "issue", "location_name_interviewer")])

}



#-----------------------------------------------------------------------------#

# 16.2 finding duplicated rows in the data frame upto age class
```

```
# Cat: duplicated rows in hh_gender_age

# Input: hh_gender_age.csv

# Output: list of duplicated rows upto age class

#-----------------------------------------------------------------------------#

#finding the duplicated rows for hh_id and age class

gen_age[duplicated(gen_age[,c("household_hh_unique_no", "age_class")]),]


if(dim(gen_age[duplicated(gen_age[,c("household_hh_unique_no", "age_class")]),])[1]==0){

  output16.2 <- rbind(output16.1, data.frame(step = "step 16.2", cat="duplicated rows in
hh_gender_age",

                                             hh_unique_no = 99999,

                                             issue="No problems found",

                                             location_name_interviewer="Not applicable"))

}else{

  dr <- gen_age[duplicated(gen_age[,c("household_hh_unique_no", "age_class")]),] #dr = duplicated
rows

  dr$step <- "Step 16.2"

  dr$cat <- "duplicated rows in hh_gender_age"

  dr$issue <- "row duplicated up to age class"

  dr <- merge(dr, hh_info[,c("hh_unique_no",

                             "location_name_interviewer")],

              by.x = "household_hh_unique_no",

              by.y = "hh_unique_no",

              all.x=T)

  names(dr)[1]<- "hh_unique_no"

  output16.2<- rbind(output16.1, dr[,c("step", "cat",

                                       "hh_unique_no",

                                       "issue", "location_name_interviewer")])

}
```

```
#------------------------------------------------------------------------------#

# 16.3 finding mismatched hh_id in hh_gender_age with primary_tree_forest_products.csv

# Cat: missmatch of hh_id btwn hh_gender_age and primary_tree_forest_products

# Input: hh_gender_age.csv and primary_tree_forest_products.csv

# Output: list of mismatched hh_id

#------------------------------------------------------------------------------#

#hh_id not in sell list of primary_tree_forest_products.csv

test1<- gen_age$household_hh_unique_no[!(gen_age$household_hh_unique_no %in%
for_prodt$household_hh_unique_no)]

#hh_id not in gen_age_pdt.csv

test2 <- for_prodt$household_hh_unique_no[!(for_prodt$household_hh_unique_no %in%
gen_age$household_hh_unique_no)]

test <- data.frame(hh_unique_no = union(test2, test1))



if(dim(data.frame(hh_unique_no = union(test2, test1)))[1]==0){

  output16.3 <- rbind(output16.2, data.frame(step = "step 16.3", cat="mismatch betwn hh_gender_age
and for_prodt",

                                             hh_unique_no = 99999,

                                             issue="No problems found",

                                             location_name_interviewer="Not applicable"))

}else{

  test <- data.frame(hh_unique_no = union(test2, test1))

  test$step <- "Step 16.3"

  test$cat <- "mismatch betwn hh_gender_age and for_prodt"

  test$issue <- ifelse(test$hh_unique_no==test1, "hh_id missing in primary_tree_forest_products",

                       "hh_id missing in hh_gender_age")

  test <- merge(test, hh_info[,c("hh_unique_no",
```

```
                        "location_name_interviewer")],
            by = "hh_unique_no", all.x=T)
  output16.3 <- rbind(output16.2, test[,c("step", "cat",
                                "hh_unique_no",
                                "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------#
# 16.4 The number of hh members involved in primary tree and forest product
#      collection are the subset of total hh member. The number of hh member
#      in hh_gender_age shouldn't exceed  the total hh member in household.
#      So, under this check-
#      "the number of hh members in hh_gender_age is compared with the total
#      hh members from household.csv"
# Cat: inconsistency in hh members of hh_gender_age
# Input: hh_gender_age.csv and household.csv
# Output: list of hh_id with greater hh members in hh_gender_age than household.csv
#-------------------------------------------------------------------------#
test16.4 <- gen_age
test16.4[,4:6][is.na(test16.4[,4:6])] <- 0
test16.4$tot_hh_mem <- test16.4$male+test16.4$female+test16.4$third_gender
test16.4 <- aggregate(data=test16.4, tot_hh_mem~household_hh_unique_no, FUN=sum)
test16.4 <- merge(test16.4, hh_info[,c("hh_unique_no", "demographic_info_hh_members")],
              by.x = "household_hh_unique_no",
              by.y="hh_unique_no", all.x=T)
test16.4 <- test16.4[test16.4$tot_hh_mem > test16.4$demographic_info_hh_members,]

if(dim(test16.4[test16.4$tot_hh_mem > test16.4$demographic_info_hh_members,])[1]==0){
```

```
 output16.4 <- rbind(output16.3, data.frame(step = "step 16.4", cat="inconsistency in hh members of
hh_gender_age",
                                  hh_unique_no = 99999,
                                  issue="No problems found",
                                  location_name_interviewer="Not applicable"))
}else{
 test16.4 <- test16.4[test16.4$tot_hh_mem > test16.4$demographic_info_hh_members,]
 test16.4$step <- "Step 16.4"
 test16.4$cat <- "inconsistency in hh members of hh_gender_age"
 test16.4$issue <- "hh members in hh_gender_age > hh members in household"
 test16.4 <- merge(test16.4, hh_info[,c("hh_unique_no",
                                  "location_name_interviewer")],
           by.x = "household_hh_unique_no",
           by.y = "hh_unique_no",
           all.x=T)
 names(test16.4)[1]<- "hh_unique_no"
 output16.4<- rbind(output16.3, test16.4[,c("step", "cat",
                                  "hh_unique_no",
                                  "issue", "location_name_interviewer")])
}



#-----------------------------------------------------------------------------#
# 17.1 Data in hh_tree_emp is about employment in tree and forest related
#      activities. There is chanche of missing data in some column which
#      must have meaningful data.
#      So the check is about-
#      "data missing in some columns of hh_tree_emp
# Cat: missing data in hh_tree_emp
```

```
# Input: hh_tree_emp.csv
# Output: list of cells missing employment data
#-----------------------------------------------------------------------------#
dm <- filter(tree_emp, hh_gender==0|is.na(hh_gender)| #dm = data missing
             hh_age_emp==0|is.na(hh_age_emp)|mnth_emp==0|is.na(mnth_emp)|
             hh_emp_salary==0|is.na(hh_emp_salary))

if(dim(filter(tree_emp, hh_gender==0|is.na(hh_gender)|
            hh_age_emp==0|is.na(hh_age_emp)|mnth_emp==0|is.na(mnth_emp)|
            hh_emp_salary==0|is.na(hh_emp_salary)))[1]==0){
  output17.1 <- rbind(output16.4,
                  data.frame(step = "step 17.1",
                          cat="missing data in hh_tree_emp",
                          hh_unique_no = 99999,
                          issue="No problems found",
                          location_name_interviewer="Not applicable"))
}else{
  dm <- filter(tree_emp, hh_gender==0|is.na(hh_gender)|
              hh_age_emp==0|is.na(hh_age_emp)|mnth_emp==0|is.na(mnth_emp)|
              hh_emp_salary==0|is.na(hh_emp_salary))
  dm$step <- "Step 17.1"
  dm$cat <- "missing data in hh_tree_emp"
  dm$issue <- ifelse(dm$hh_gender==0 | is.na(dm$hh_gender),
                  "hh_gender data missing",
                  ifelse(dm$hh_age_emp==0|is.na(dm$hh_age_emp),
                       "hh_age_emp data missing",
                       ifelse(dm$mnth_emp==0|is.na(dm$mnth_emp),
                            "mnth_emp data missing",
                            "hh_emp_salary data missing")))
```

```
 dm <- merge(dm, hh_info[,c("hh_unique_no",
                      "location_name_interviewer")],
          by.x = "household_hh_unique_no",
          by.y = "hh_unique_no",
          all.x=T)
 names(dm)[1]<- "hh_unique_no"
 output17.1<- rbind(output16.4, dm[,c("step", "cat",
                              "hh_unique_no",
                              "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------#
# 17.2 Sometimes the enumerators entered same data twice or more. From that
#     standpoint, there is chance to exist duplicated row upto certain
#     level.
#     So, the check is about to-
#     "find duplicated rows in the data frame upto hh_age_emp
# Cat: duplicated rows in hh_tree_emp
# Input: hh_tree_emp.csv
# Output: list of duplicated rows upto age of employed hh member
#-------------------------------------------------------------------------#
#finding the duplicated rows for hh_id, hh_gender, hh_age_emp
tree_emp[duplicated(tree_emp[,c("household_hh_unique_no", "hh_gender",
                        "hh_age_emp", "hh_emp_salary")]),]


if(dim(tree_emp[duplicated(tree_emp[,c("household_hh_unique_no", "hh_gender",
                              "hh_age_emp", "hh_emp_salary")]),])[1]==0){
 output17.2 <- rbind(output17.1,
```

```
                data.frame(step = "step 17.2",
                           cat="duplicated rows in hh_tree_emp",
                           hh_unique_no = 99999,
                           issue="No problems found",
                           location_name_interviewer="Not applicable"))
}else{
  dr <- tree_emp[duplicated(tree_emp[,c("household_hh_unique_no", "hh_gender",
                                        "hh_age_emp","hh_emp_salary")]),]
  dr$step <- "Step 17.2"
  dr$cat <- "duplicated rows in hh_tree_emp"
  dr$issue <- paste("hh position", dr$X_hh_tree_emp_position,
                    "duplicated upto money spent for energy")
  dr <- merge(dr, hh_info[,c("hh_unique_no",
                             "location_name_interviewer")],
              by.x = "household_hh_unique_no",
              by.y = "hh_unique_no",
              all.x=T)
  names(dr)[1]<- "hh_unique_no"
  output17.2<- rbind(output17.1, dr[,c("step", "cat",
                                       "hh_unique_no",
                                       "issue", "location_name_interviewer")])
}



#-----------------------------------------------------------------------------#
# 17.3 The salary of the employed hh members seems very high in some cases
#      which may not be consistent with the age and nature of job.
#      So, the check is about to-
#      "find outliers in salary of the employed hh members"
```

```
# Cat: salary outliers in hh_tree_emp
# Input: hh_tree_emp.csv
# Output: list of hh_id having outliers in hh_emp_salary
#-----------------------------------------------------------------------------#
if(dim(tree_emp[tree_emp$hh_emp_salary>50000|
            (tree_emp$hh_age_emp>60&tree_emp$mnth_emp==12 &
             tree_emp$hh_emp_salary>15000),])[1]==0){
  output17.3 <- rbind(output17.2, data.frame(step = "step 17.3",
                                  cat="salary outliers in hh_tree_emp",
                                  hh_unique_no = 99999,
                                  issue="No problems found",
                                  location_name_interviewer="Not applicable"))
}else{
  outlier <- tree_emp[tree_emp$hh_emp_salary>50000|
                  (tree_emp$hh_age_emp>60 & tree_emp$mnth_emp==12 &
                    tree_emp$hh_emp_salary>15000),]
  outlier$step <- "Step 17.3"
  outlier$cat <- "salary outliers in hh_tree_emp"
  outlier$issue <- ifelse(outlier$hh_emp_salary>50000,
                    paste("salary too high:",outlier$hh_emp_salary,"TK"),
                    paste("age >60 yr but works 12 mnths and earns",
                        outlier$hh_emp_salary,"TK"))
  outlier <- merge(outlier, hh_info[,c("hh_unique_no",
                               "location_name_interviewer")],
              by.x = "household_hh_unique_no", by.y = "hh_unique_no", all.x=T)
  names(outlier)[1] <- "hh_unique_no"
  output17.3 <- rbind(output17.2, outlier[,c("step", "cat",
                                  "hh_unique_no",
                                  "issue", "location_name_interviewer")])
```

```
}



#-----------------------------------------------------------------------------#
# 18.1 Other_income_sources table contains hh income from sources other than
#      trees and forests. There is chance ofmissing data in columns which
#      must contain data.
#      So, the check is about to-
#      identify the missing data in other_income_sources particularly in
#      other_sources_type, income_other_sources columns
# Cat: missing data in othr_income
# Input: other_income_sources.csv
# Output: list of hh_id missing income data from sources other than trees
#         and forests
#-----------------------------------------------------------------------------#
dm <- filter(othr_income, is.na(other_sources_type)|other_sources_type==0|
             is.na(income_other_sources)|income_other_sources==0) #dm = data missing


if(dim(filter(othr_income, is.na(other_sources_type)|other_sources_type==0|
            is.na(income_other_sources)|income_other_sources==0))[1]==0){
  output18.1 <- rbind(output17.3, data.frame(step = "step 18.1",
                                     cat="missing data in othr_income",
                                     hh_unique_no = 99999,
                                     issue="No problems found",
                                     location_name_interviewer="Not applicable"))
}else{
  dm <- filter(othr_income, is.na(other_sources_type)|other_sources_type==0|
               is.na(income_other_sources)|income_other_sources==0)
  dm$step <- "Step 18.1"
```

```
 dm$cat <- "missing data in othr_income"
 dm$issue <- ifelse(is.na(dm$other_sources_type)|dm$other_sources_type==0,
                "data missing in income source type",
                "data missing in income amnt")
 dm <- merge(dm, hh_info[,c("hh_unique_no",
                      "location_name_interviewer")],
          by.x = "household_hh_unique_no",
          by.y = "hh_unique_no",
          all.x=T)
 names(dm)[1]<- "hh_unique_no"
 output18.1<- rbind(output17.3, dm[,c("step", "cat",
                              "hh_unique_no",
                              "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------------#
# 18.2 There is change of doubel entry of same data by the enumerator.
#      Duplicacy may occur at different levles/columns.Identifying dulicacy
#      upto some basic level is expected to help checking this kind of error.
#      So, this check is about to-
#      "find the duplicated rows in other_income_sources data up to
#      other_sources_type"
# Cat: duplicated rows in other_income_sources
# Input: other_income_sources.csv
# Output: list of duplicated rows upto other_sources_type
#-------------------------------------------------------------------------------#
#finding the duplicated rows for hh_id and age class
othr_income[othr_income$other_sources_type!=96,][duplicated(othr_income[
```

```
  othr_income$other_sources_type!=96,][,c("household_hh_unique_no",
                                  "other_sources_type")]),]

if(dim(othr_income[othr_income$other_sources_type!=96,][duplicated(
  othr_income[othr_income$other_sources_type!=96,][,c("household_hh_unique_no",
                                       "other_sources_type")]),])[1]==0){
  output18.2 <- rbind(output18.1, data.frame(step = "step 18.2",
                                  cat="duplicated rows in other_income_sources",
                                  hh_unique_no = 99999,
                                  issue="No problems found",
                                  location_name_interviewer="Not applicable"))
}else{
  dr <- othr_income[othr_income$other_sources_type!=96,][duplicated(othr_income[
    othr_income$other_sources_type!=96,][,c("household_hh_unique_no",
                                  "other_sources_type")]),] #dr = duplicated rows
  dr$step <- "Step 18.2"
  dr$cat <- "duplicated rows in other_income_sources"
  dr$issue <- paste("hh position", dr$X_other_income_sources_position,
                "duplicated upto other_sources_type")
  dr <- merge(dr, hh_info[,c("hh_unique_no",
                         "location_name_interviewer")],
              by.x = "household_hh_unique_no",
              by.y = "hh_unique_no",
              all.x=T)
  names(dr)[1]<- "hh_unique_no"
  output18.2<- rbind(output18.1, dr[,c("step", "cat",
                                  "hh_unique_no",
                                  "issue", "location_name_interviewer")])
}
```

```
#-------------------------------------------------------------------------------#
# 18.3 Other_income_sources data file contains the partial data on other
#      income sources of the surveyed households. The basic data i.e. whether
#      the hh receive income from sources other than trees and  forests,
#      how much income received etc. are compiled in the household.csv.
#      Details of the other income sources of those hh is compiled in the
#      other_income_sources data file.
#      So, this check is about to-
#      "find mismatching hh_id in other_income_sources and household"
# Cat: missmatch of hh_id between other_income and household
# Input: other_income_sources.csv and household.csv
# Output: list of mismatched hh_id
#-------------------------------------------------------------------------------#
#hh_id not in household.csv
test1<- othr_income$household_hh_unique_no[!(othr_income$household_hh_unique_no %in%
hh_info[hh_info$economics_livelihood_hh_income_other==1,]$hh_unique_no)]
#hh_id not in other_income_sources.csv
test2 <-
hh_info[hh_info$economics_livelihood_hh_income_other==1,]$hh_unique_no[!(hh_info[hh_info$econ
omics_livelihood_hh_income_other==1,]$hh_unique_no %in%
othr_income$household_hh_unique_no)]
test <- data.frame(hh_unique_no = union(test2, test1))



if(dim(data.frame(hh_unique_no = union(test2, test1)))[1]==0){
  output18.3 <- rbind(output18.2, data.frame(step = "step 18.3",
                                             cat="missmatch of hh_id btwn other_income and household",
                                             hh_unique_no = 99999,
```

```
                                issue="No problems found",
                                location_name_interviewer="Not applicable"))
}else{
  test <- data.frame(hh_unique_no = union(test2, test1))
  test$step <- "Step 18.3"
  test$cat <- "missmatch of hh_id btwn other_income and household"
  test$issue <- "hh_id mismatch between other_income and household"
  mismatch <- merge(test, hh_info[,c("hh_unique_no",
                              "location_name_interviewer")],
                    by = "hh_unique_no", all.x=T)
  output18.3 <- rbind(output18.2, mismatch[,c("step", "cat",
                                  "hh_unique_no",
                                  "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------------#
# 18.4 As described in the check 18.3 that the total income from other sources
#      for each hh should be close to the total income from other sources
#      mentioned in the househld.csv.
#      So, this check is about to-
#      "compare the total hh income from other_income_sources with the
#      total hh income from other sources mentioned in household.csv"
# Cat: inconsistency in hh total income from other sources
# Input: other_income_sources.csv and household.csv
# Output: list of hh_id with higher differences in other sources income from
#     other_incoem_sources.csv and household.csv
#-------------------------------------------------------------------------------#
test18.4 <- othr_income
```

```
test18.4$income_other_sources <- ifelse(is.na(test18.4$income_other_sources),
                                        0, test18.4$income_other_sources)
test18.4 <- aggregate(data=test18.4, income_other_sources~household_hh_unique_no, FUN=sum)
test18.4 <- merge(test18.4, hh_info[,c("hh_unique_no", "economics_livelihood_income_other_tree")],
                  by.x = "household_hh_unique_no",
                  by.y="hh_unique_no", all.x=T)
test18.4$diff_income <- abs(test18.4$income_other_sources -
test18.4$economics_livelihood_income_other_tree)


if(dim(test18.4[test18.4$diff_income>0,])[1]==0){
  output18.4 <- rbind(output18.3, data.frame(step = "step 18.4", cat="inconsistency in hh total income
from other sources",
                                             hh_unique_no = 99999,
                                             issue="No problems found",
                                             location_name_interviewer="Not applicable"))
}else{
  incon_income <- test18.4[test18.4$diff_income>0,]
  incon_income$step <- "Step 18.4"
  incon_income$cat <- "inconsistency in hh total income from other sources"
  incon_income$issue <- paste("income from other sources is different in other_income_sources and
household:",incon_income$diff_income)
  incon_income <- merge(incon_income, hh_info[,c("hh_unique_no",
                                                 "location_name_interviewer")],
                        by.x = "household_hh_unique_no",
                        by.y = "hh_unique_no",
                        all.x=T)
  names(incon_income)[1]<- "hh_unique_no"
  output18.4<- rbind(output18.3, incon_income[,c("step", "cat",
                                                 "hh_unique_no",
                                                 "issue", "location_name_interviewer")])
```

```
}


#-------------------------------------------------------------------------------#
# 18.5 Somtiems the income sources under 96 category is not specified by mistake.
#      So, this check is about to-
#      "find the hh id with unspecified 96 category of other income sources
# Cat: Other income source category 96 not specified
# Input: other_income_sources.csv
# Output: list of hh_id with unspecified 96 category of other income sources
#-------------------------------------------------------------------------------#
prob_96 <- othr_income[othr_income$other_sources_type==96 &
                       othr_income$other_sources_type_qualifier=="",]
if(dim(othr_income[othr_income$other_sources_type==96 &
                   othr_income$other_sources_type_qualifier=="",])[1]==0){
  output18.5 <- rbind(output18.4, data.frame(step = "step 18.5",
                                             cat="Other income source category 96 not specified",
                                             hh_unique_no = 99999,
                                             issue="No problems found",
                                             location_name_interviewer="Not applicable"))
}else{
  prob_96 <- othr_income[othr_income$other_sources_type==96 &
                           othr_income$other_sources_type_qualifier=="",]
  prob_96$step <- "Step 18.5"
  prob_96$cat <- "Other income source category 96 not specified"
  prob_96$issue <- "Other income source category 96 but source not specified"
  prob_96 <- merge(prob_96, hh_info[,c("hh_unique_no",
                                       "location_name_interviewer")],
                   by.x = "household_hh_unique_no",
                   by.y = "hh_unique_no",
```

```
                    all.x=T)
  names(prob_96)[1]<- "hh_unique_no"
  output18.5<- rbind(output18.4, prob_96[,c("step", "cat",
                                  "hh_unique_no",
                                  "issue", "location_name_interviewer")])
}



#-------------------------------------------------------------------------#
# 18.6 The other income sources under 96 category were written differetly
#      by the enumerators which are mistaken by spelling and not uniform by
#      name.
#      SO, this check is about to-
#      "compile the non-uniform spelling and names"
# Cat: non-uniform spelling under 96 income source category
# Input: other_income_sources.csv
# Output: list of non-uniform spelling hh_id
#-------------------------------------------------------------------------#
income_96 <-
sort(unique(othr_income[othr_income$other_sources_type==96,]$other_sources_type_qualifier))
income_96 <- paste(as.character(income_96), collapse="; ")


output18.6 <- rbind(output18.5, data.frame(step = "step 18.6",
                                  cat="non-uniform spelling under 96 income category",
                                  hh_unique_no = 99999,
                                  issue=paste("These (", distb_96, ") need to be of uniform spelling", sep = ""),
                                  location_name_interviewer="Not applicable"))
```

#----------#

19 hh_combined is a very large data file containing all the basic and important hh data. Following are some observation on different column. The observations are provided by columns. Since a lot of QC issues are associated with this file which will take substantial time for coding in r, so QC observations provided as comments by column.

input: household_combined_2nd.csv checks

Output: (Columns: Comments)

#----------#

location_forest_admin_forest_div:forest division name missing; there are some meaning less numbers

location_forest_admin_forest_rang: missing

location_forest_admin_forest_beat: missing

location_hh_location_point: data missing (hh id 299)

location_hh_location_gps_x: (hh_id 3872 and 4862) have 2 values (9123344 and 9.06E+11) need to be checked

location_hh_location_gps_y: (hh_id 4508 and 4893) have 2 values (2413326 and 2.38E+09) that need to be checked

location_start_time_hour and location_start_time_minute missing (hh_id 4046)

demographic_info_name_respondent missing (hh_id 4955)

demographic_info_name_respondent name (Ã???haduzzaman) not clear (hh_id 927)

demographic_info_hh_ethnicity_qualifier: i) Bengali under 96 category; ii) names under 96 should be checked and uniformly spelled i.e. telugo etc.

demographic_info_hh_members: i) hh members too high (>15) (hh_id 1907, 1935, 1940, 5403, 2507, 2745, 2784, 4303, 4375)

biodiversity_conservation_plant_spp: Answer:No- No plant species (that was abundant earlier) became rare but species names provided (hh_id: 1793, 2937, 2940) and reduction reason mentioned

biodiversity_conservation_plant_abundant[1-5]: Many of the local names are uncommon. I think some names are misspelled and some are very localized which will make it difficult to trace the original plant

biodiversity_conservation_reason_plant_reduct[1-5]: Enumerators written 5 instead of 96 as other reasons

biodiversity_conservation_reason_plant_reduct_qualifier[1-5]: The reason qualifier for 96 category is uniform in most of the cases. In few cases same names are little deviated which will create two results for same name during estimation i.e. i) Salinity written in several ways: "salinity"; "silinity"; "Silinity,"; "Sailinity"; "sÃ linity"; ii) Sometims two reasons are written in one cell: "Silinity,waterlogging"; salinity/

clamate change ; iii) some reasons are not clear: tower; iv) most of the reasons are in english which is fine but some reasons are written in bengali; v) all the names need to be checked thoroughly etc.

biodiversity_conservation_plant_spp_common: In some cases response is "0" but species name and reason is mentioned, similarly response is "1" (hh_id 4597) but plant name or reason not mentioned; response "99" (hh 4601, 4707) but reason mentioned

biodiversity_conservation_plant_abundant_comm[1-5]: Some plant names are not clear may be due to very localized name or misspelling; some names still need to be revised i.e. i) "Akashmoni"; "Akashi"; "Akadhi" ;ii) "Gawya"; "Gaowa"; All the names need throughly checked again to make more uniform (spelling and language) and meaningful

biodiversity_conservation_reason_plant_increase_qualifier[1-5]: comment is same as biodiversity_conservation_reason_plant_reduct_qualifier[1-5]

biodiversity_conservation_tree_cover_past: response is "2" but mentioned reason of decrease (answered 8.3.3.5) which should be blank

biodiversity_conservation_tree_cover_increase[1-5]: i) the reasons need to check again to make more uniform i.e. "Less salinity"; "Less salinitu" (hh 716); ii) some names are not clear i.e. Less salinitu "La"; "Ov" (hh 6001); iii) response for the reason is 2

#

#export SE output

write.csv(output18.6, file="./Outputs/se_qc_output_Oct.csv", row.names = F)