Useful functions for PIR15

Name	**Content**	**Introduce external library if needed***
read_csv(file_path)	Read a .csv file	library(readr)
write_csv(variable, file_path)	Write a variable to a .csv file	library(readr)
setwd(path)	Set the defined workspace directory	
subset (df, condition)	Choose a part of dataset df, which satisfies the condition, e.g. subset(df, device ==”Device 104”)	
which(condition)	Return the index (position) of variable, satisfying the true condition; e.g. df[which(df$device ==”Device 104”),]; df[which(df$device %in% c(”Device 104”, “Device 1060)),]	
with(df, field)	Equivalent to df$field	
order(field, decreasing = FALSE**)**	Sequence order as the field by decreasing (TRUE), or increasing (FALSE); e.g. df[with(df,order(deb_H, decreasing = FALSE)),]	
unique(field_1)	Remove duplicated items in field_1; e.g. length(unique(df$device))	
table(field_1) **table**(field_1, field_2)	Obtain the frequency of occurrence of items in field 1; or cross matrix of field_1 and field_2; Attention: the field type of “factor” can keep all levels of factors for the statistic even only one item value is selected for that. e.g. rowSums(table(df$device, df$jour)!=0)	

as.character(field) as.numeric(field) as.factor(field) **as.data.frame**(table)	Convert the field type to “charcter”,”numeric”,”factor”, etc Or convert table/matrix to data.frame	
names(df) **rownames(**table**)** **colnames(**table**)**	Show the field names of df; can also modify the filed name with assigned values (character)	
merge(df1, df2, by.x , by.y, all.x, all.y)	Combine two dataframes together; e.g. **merge**(df1, df2, by.x = “field_1”, by.y = “field_2”, all.x = TRUE, all.y = FALSE), means that we merge df1 and df2 into one data.frame by the index of field_1 in df1 and the index of field_2 in df2 (or more than one field each), and should keep all index of field_1, rather than all in field_2.	
aggregate(df, list(df$field1), fun) **aggregate**(df, by=list(df$field1, df$field2), fun)	Aggregate the fields in df by the groups of field 1 (or field 1 and field 2), to get the sum or mean value by setting ‘ fun’ as sum or mean, respectively.	
lapply(x, FUN**)**	Arguments: -x: A vector or an object -FUN: Function (self-defined) applied to each element of x	
%>%	successive execution; e.g. Nb_cells <- lapply(min_dur_sen, NbCellFun) %>% bind_rows() # Function Return df and combine all dfs by rows	library(dplyr)
ggplot()	Plot function set	library(ggplot2)

Note* : if no specified package, install it firstly, then code *library (name)* to introduce it.