Data Manipulation-02 Handling Data in dplyr

Created by H. M. Samadhi Chathuranga Rathnayake

```
install.packages("dplyr")#Install only if it has not been installed
library(dplyr)

setwd("D:\\Workshops\\R Programming for Data Science Workshop\\Part 02 - Data
Manipulation & Cleaning\\Datasets")
data=read.csv("iris.CSV")

colnames(data)

#Selecting columns
df=select(data,Sepal.Length)
head(df)

df=select(data,Sepal.Length,Petal.Width)
head(df)

df=select(data,Species,Petal.Width)
head(df)

df=select(data,c(Sepal.Length,Petal.Width,Id))
head(df)

df=select(data,-Sepal.Length,-Petal.Width)
head(df)

df=select(data,-c(Sepal.Length,Petal.Width))
head(df)

df=select(data,Sepal.Length:Petal.Width)
head(df)

df=select(data,3)
head(df)

df=select(data,1,2,3)
head(df)

df=select(data,c(1,2,3))
head(df)

df=select(data,-1,-2,-3)
head(df)

df=select(data,-c(1,2,3))
head(df)
```

```r
df=select(data,1:4)
head(df)

df=select(data,"SL"=Sepal.Length,"SW"=Sepal.Width)
head(df)

df=slice(data,1:10)
df

#Rename the columns
df=rename(data,"SL"=Sepal.Length,"SW"=Sepal.Width)
head(df)

#The same things can be done with the compound assignment operator and the
assignment operator
data %>% select(Sepal.Length,Petal.Length) ->new_data
head(new_data)

data %>% select(-Sepal.Length,-Petal.Length) ->new_data
head(new_data)

data %>% select(starts_with("S")) ->new_data
head(new_data)

data %>% select(ends_with("s")) ->new_data
head(new_data)

data %>% slice(1:7) ->new_data
new_data

data %>% rename("SL"=Sepal.Length,"SW"=Sepal.Width) ->new_data
head(new_data)

pull(data,Sepal.Length) #What this does is convert the sub column into a
vector

data %>% pull(Sepal.Length)->new_data
new_data

#Filtering columns
df=filter(data,Sepal.Length>=5)
df

df=filter(data,Sepal.Length>=5,Species=="setosa")
df

df=filter(data,Sepal.Length>=5,Species=="setosa",Sepal.Width>=4)
df

data %>% filter(Sepal.Length>=5,Species=="setosa",Sepal.Width>=4) ->new_data
new_data
```

```r
#Applying several operations together
data %>% select(Sepal.Length,Sepal.Width,Species) %>%
filter(Sepal.Length>=5,Species=="setosa",Sepal.Width>=4) ->new_data
new_data

#Convert row names to a new column
data("mtcars") #mtcars inbuilt data set is used
head(mtcars)

mtcars_new=add_rownames(mtcars,"Car_Names")

mtcars_new

#How can we do this in Base R
mtcars_new2=mtcars
mtcars_new2$Car_Names=row.names(mtcars)
head(mtcars_new2)

row.names(mtcars_new2)=1:nrow(mtcars_new2)
head(mtcars_new2)

#Creating new columns with existing columns
bmi=read.csv("BMI.CSV")
head(bmi)

bmi_new=mutate(bmi,BMI=(Weight)/((Height/100)**2))
head(bmi_new)

bmi=read.csv("BMI.CSV")
head(bmi)

bmi_new=mutate(bmi,Hieght=Height/100,BMI=(Weight)/((Height/100)**2),Height=NU
LL)
head(bmi_new)

bmi=read.csv("BMI.CSV")
head(bmi)

bmi_new=transmute(bmi,Hieght=Height/100,BMI=(Weight)/((Height/100)**2))
head(bmi_new)

marks=read.csv("marks.CSV")
head(marks)

result=mutate(marks,Result=ifelse(Marks>=50,"Pass","Fail"))
head(result)

#if_else is a function inside dplyr
result2=mutate(marks,Result=if_else(Marks>=50,"Pass","Fail"))
head(result2)

marks=sample(20:100,10000000,replace = TRUE)
system.time(ifelse(marks>=50,"Pass","Fail"))
```

```r
system.time(if_else(marks>=50,"Pass","Fail")) #if_else is much faster

#Distinct values
live=read.csv("Living_Area.CSV")
head(live)

distinct(live,Gender)

distinct(live,Living)

distinct(live,Gender,Living)

#Sorting the data frame by a column
data=read.csv("iris.CSV")
head(data)

df1=arrange(data,Sepal.Length)
head(df1)

tail(df1)

df2=arrange(data,desc(Sepal.Length))
head(df2)

tail(df2)

df3=arrange(data,Sepal.Length,Petal.Length)
head(df3)

tail(df3)

#Summaries of data
head(data)

summarise(data,mean(Sepal.Length),median(Sepal.Width),sum(Petal.Length))

#Summaries for columns with grouping by a column
groups=group_by(data,Species)
summarise(groups,mean(Sepal.Length),median(Sepal.Width),sum(Petal.Length))

#As we discussed earlier any of these functions can be written as the
following way too
data %>% summarise(mean(Sepal.Length),median(Sepal.Width),sum(Petal.Length))-
>details
details

#Getting a sample from the data frame
sample_ID=sample(1:nrow(data),20)
samdata1=data[sample_ID,]
samdata1

samdata2=data[-sample_ID,]
samdata2
```

```r
samdata3=sample_n(data,20)
samdata3

samdata4=sample_frac(data,0.5)
samdata4

#Combining data frames

#Joining data
df1=data.frame(ID=1:3,Group1=c("A","B","C"))
df2=data.frame(ID=2:4,Group2=c("P","Q","R"))

#Mutating joins (Merging data)
inner_join(df1,df2,by="ID")

left_join(df1,df2,by="ID")

right_join(df1,df2,by="ID")

full_join(df1,df2,by="ID")

#Filter joins (2nd data frame will be used as a filter for deleting in 1st
data frame)
semi_join(df1,df2,by="ID")

anti_join(df1,df2,by="ID")

#Binding rows and columns
df1=data.frame(ID=1:3,Group1=c("A","B","C"))
df2=data.frame(ID=4:6,Group1=c("P","Q","R"))
df3=data.frame(Group2=c("P","Q","R"))

bind_rows(df1,df2)

bind_cols(df1,df3)

#Dropping columns in a data frame
data=read.csv("iris.CSV")
head(data)

data %>% select(-Species)->new_data
new_data

data %>% select(-c(Id,Species))->new_data
new_data

data %>% select(-one_of(c("Id","Species")))->new_data
new_data
```