# Concepts General

## How are Bools useful?

* We can evaluate any expression in R and get one of two answers True or False
* “Comparators” (<,>, ==, !=) return a bool
* You can get the numerical value of bools in a data frame by multiplying that column or data frame by 1.
* You can also get the numerical value of bools by using **sapply()**.

## Why do we need to use a sample of a population?

* A population is an entire group that you want to draw conclusions from, a sample is a subset of that population.
* Sometimes you will not be able to have data on the entire population, so you need a representative dataset
* It is more cost effective to process a smaller subset

### What is the concept of Sampling?

Data sampling is a statistical analysis technique used to select, manipulate and analyze a representative subset of data points to identify patterns and trends in the larger data set being examined.

### How/Why do we use Sampling?

**Analyze data, describe what was found, infer for future batches**

## Compare basic concepts of descriptive statistics

* + **Mean** 
    - The average of a dataset
  + **Median**
    - The middle value/number within a data set
  + **Mode**
    - The most common value/number within a dataset
  + **Range**
    - The difference between the largest and smallest values
  + **Variance**
    - Spread between numbers in a dataset
    - Measures how far each number in the data set is from the mean and thus every other number in the set
    - Denoted by the symbol: σ2
  + **Standard Deviation**
    - Measure of Dispersion (variability) of data in relation to the mean
    - The closest to zero the standard deviation is the more close to the mean the values in the studied dataset are.

## Why do we want to know Mean vs. Median?

* Both are a measure of **“Central Tendency”** within a dataset, but often are two different values.
* Mean is most often used as a general measure of central tendency; however, a data scientist will want to use median when
  + There are a few extreme scores within the distribution of the data, skewing of the data.
  + There are missing or undetermined values within the data
  + An open-ended distribution, where an option may be 1,2,3,**4+**
  + Data is measured within an **Ordinal Scale**
    - Ordinal Scale – scale of measurement that uses labels to classify cases (measurements) into ordered classes, ex. Movie ratings 5/5 Military ranks E1 to E-9 basically for movie reviews the difference between a 4-star rating and a 5-star rating cannot be quantified.

## How do we gain information about a dataset?

### Process

First we want to learn as much about a data set as we can at a glance:

1. How big is the data?
   1. How many columns and rows are we working with?
2. What is the range of the data?
3. What data types are we working with?
4. How is the data distributed?

### Methods for getting information about data set/dataframe

* **head()**  Function
  + Allows us to view the 1st 6 rows by default
  + Text

    Description automatically generated
* **tail()** function
  + Allows us to view the last 6 rows
  + Text

    Description automatically generated
* **dim()** Function
  + Allows us to view the dimensions of the data frame. This returns [rows, columns]
  + 
* **dimnames()** Function
  + Retrieve or set the dimension names of an R object
  + A picture containing text, receipt

    Description automatically generated
* **summary()** Function
  + Summary will show each column, it’s data type, and other attributes
  + It will also display the 1st quartile, median, mean 3rd quartile and max values
  + A picture containing text, receipt

    Description automatically generated
* **str()** Function
  + Displays the internal **str**ucture of an R object, an alternative to ‘summary’
  + Text

    Description automatically generated

## Data Models/Data Frames

## Data Cleaning Why do we do it?

* We need data formatted to work with it properly and accurately
* Prevention and removal of errors
* Irrelevant and corrupted data should be removed
* Formatted and arranged for processing

## What is Descriptive Statistics?

A descriptive statistic is a summary statistic that quantitatively describes or summarizes features from a collection of information, while descriptive statistics is the process of using and analyzing those statistics.

## Central Limit Theorem

**States that the distribution of a sample variable approximates a normal distribution** (i.e., a “bell curve”) as the sample size becomes larger, **assuming that all samples are identical in size**, and regardless of the population's actual distribution shape.

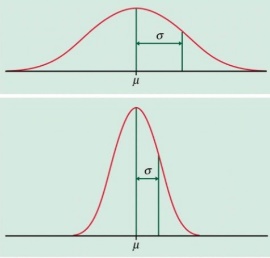
Put another way, CLT is a **statistical premise that, given a sufficiently large sample size** from a population with a finite level of variance, **the mean of all sampled variables from the same population** **will be approximately equal to the mean of the whole population**.

## Distributions of Means

The distribution of sample means is defined as the set of means from all the possible random samples of a specific size (n) selected from a specific population.

## Standard Deviation

A standard deviation (or σ) is a measure of how dispersed the data is in relation to the mean. Low standard deviation means data are clustered around the mean, and high standard deviation indicates data are more spread out.



## Replication

The default behaviour is as if the call was

rep(x, times = 1, length.out = NA, each = 1)

. Normally just one of the additional arguments is specified, but if each is specified with either of the other two, its replication is performed first, and then that implied by times or length.out.

## Quantiles

A picture containing text

Description automatically generated

Chart, box and whisker chart

Description automatically generated

## Skewness

In simple words, skewness is the measure of how much the probability distribution of a random variable deviates from the normal distribution.

## Law of Large Numbers

The law of large numbers, in probability and statistics, states that as a sample size grows, its mean gets closer to the average of the whole population.

# Read Write Code General

### Variable Assignment

**“<-“** is the assignment operand



## Data frame

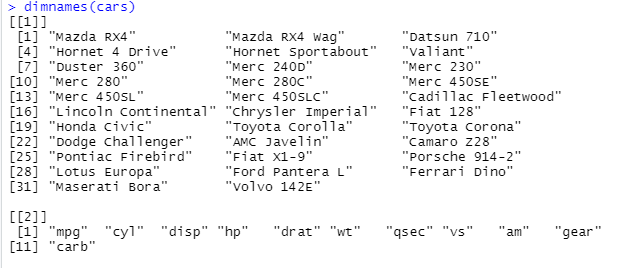
### Create data frame

Text

Description automatically generatedUse the “**data.frame()**” function to combine multiple vectors

### Combining Data

### Data frame headers



## Sorting Data Order()

### Ascending

# sort by mpg  
newdata <- mtcars[order(mpg),]  
  
# sort by mpg and cyl  
newdata <- mtcars[**order(mpg, cyl)**,]

### Descending

#sort by mpg (ascending) and cyl (descending)  
newdata <- mtcars[**order(mpg, -cyl)**,]

### Sort/order Data Frame based on column

**<dataframe>[order(dataframe$column),]**

\*\*This will return an ordered dataframe, however it will not persist

A picture containing text

Description automatically generated

## Combining Data

### Adding Columns

**merge()**

# merge two data frames by ID  
total <- merge(data frameA,data frameB,by="ID")

# merge two data frames by ID and Country  
total <- merge(data frameA,data frameB,by=c("ID","Country"))

### Adding Rows

**Rbind()**

total <- rbind(data frameA, data frameB)

## Rows

### Df.Removing Rows

removeSpecific row

dataframename = dataframename[-1,]

remove a series from index to ending

datatablename <- datatablename[2:nrow(datatablename), ]

A picture containing diagram

Description automatically generated

### Get Number of Rows **nrow()**

**nrow(<dataframe>)**

### Get Row Name(s) **rownames()**

Text

Description automatically generated **Rownames(<dataframe)**

### Get row name of specific index **rownames(cars)[1]**

Text

Description automatically generated

### Get entire row from data frame

**<dataframe>[row, column]**

Text

Description automatically generated

## Columns

### Get column names

**colnames(<dataframe>)**



### Get number of columns

**ncol(<dataframe>)**



### Print specific column

**<dataframe>[row, column]**

Text, letter

Description automatically generated

## General Functions

### Create a Boolean

Logo

Description automatically generated

A picture containing text

Description automatically generated

### Filter()

Use **filter()** to find rows/cases where conditions are true, rows where condition evaluates to NA are dropped.

### Which() Function

Will return the position of the elements (row/column number array index) where in a logical vector which are TRUE



Text

Description automatically generated

### Which.Max()

Which.Max() will return the row index of the value

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

### Min() Max()

max/min(<dataframe>) will give the max value for the entire dataframe

max/min(<dataframe>$column) will give the max value for the column only

Text

Description automatically generated

### Vector Math

You can straight up add to a vector of any type, but it will only return the vector with updated values, it will not modify the original vector.

Text

Description automatically generated

### Sum() A picture containing text Description automatically generated

### ReadCSV()

By default read.csv()reads a comma separated file and then **gives the output as a data frame**

Separator is a comma ‘,’

Arguments are:

* **Header** – TRUE by default, when true, the first row in the CSV file is set as header information (column names)
* **StringsAsFactors –** When working with categorical data (limited number or values “categories”), default behavior of R is to convert char string into factors, this makes it hard to replace values. The following example shows 3 different levels.

Text

Description automatically generated

A picture containing text

Description automatically generated

### ReadCSV2()

Nearly identical to readcsv() except:

**Separator is semi-colon ‘;’**

### Hist()

The generic function **hist** computes a histogram of the given data values. If **plot = TRUE**, the resulting object of [class](https://www.rdocumentation.org/link/class?package=graphics&version=3.6.2) **"histogram"** is plotted by [plot.histogram](https://www.rdocumentation.org/link/plot.histogram?package=graphics&version=3.6.2), before it is returned.

Text

Description automatically generated

### Unique()

Unique Values, Unique Counts, tabulate



### Function()

creates a new function

Text

Description automatically generated

### return()

returns a value of the function

### tabulate()

counts occurrences of interger, the number of bins corresponds to the number of values you want to check, 10 bins is 10 different integers

Text

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence

### Numberize a function

### mfv()

Most frequent value

### match()

**Returns the values that are in two separate lists**

Factors, raw vectors and lists are converted to character vectors, and then x and table are coerced to a common type (the later of the two types in R's ordering, logical < integer < numeric < complex < character) before matching. If incomparables has positive length it is coerced to the common type.

Matching for lists is potentially very slow and best avoided except in simple cases.

Exactly what matches what is to some extent a matter of definition. For all types, NA matches NA and no other value. For real and complex values, NaN values are regarded as matching any other NaN value, but not matching NA, where for complex x, real and imaginary parts must match both (unless containing at least one NA).

### testFrame()

### readcensus()

### var() – Variance

**statistical measurement of the spread between numbers in a data set.** More specifically, variance **measures how far each number in the set is from the mean** and thus from every other number in the set. Variance is often depicted by this symbol: σ2

var(<numeric vector, matrix or dataframe>, <default null>, na.rm <logical should missing values be removed?>)

var(x, y = NULL, na.rm = FALSE, use)

### sd() - Standard Deviation

A standard deviation (or σ) is a measure of how dispersed the data is in relation to the mean. Low standard deviation means data are clustered around the mean, and high standard deviation indicates data are more spread out

Text

Description automatically generated with medium confidence

Chart, histogram

Description automatically generated

### rnorm() - Normal Distribution

Random generation for the normal distribution with mean equal to mean and standard deviation equal to sd.

rnorm(n, mean = 0, sd = 1)

### rep() – Replicate

rep(x, times = 1, length.out = NA, each = 1)

replicate(number,mean(sample(dataframe,+size=8,replace=TRUE)),simplify=TRUE)

### sample() – Sample with rep()

sample takes a sample of the specified size from the elements of x using either with or without replacement.

Replacement is either removing the sample taken, or putting them back in the ‘basket’.

sample(x, size, replace = FALSE, prob = NULL)

Text

Description automatically generated

### mode() - Data type & mode of object

Mode will tell you what data type the column or car is

A picture containing text

Description automatically generated

### As.numeric()

Most useful for bools, with char array’s we’ll want to use as.factor()

Text, letter

Description automatically generated

### NA Values

Missing Data is represented by NA (Not available). Impossible values are also represented by NA (also known as NaN).

### Na.omit()

Text

Description automatically generated

### gsub() Pattern Matching and Replacement

**gsub(<what you want be be put in>, <what you want to replace>, <data>)**

gsub(pattern, replacement, x, ignore.case = FALSE, perl = FALSE, fixed = FALSE, useBytes = FALSE)

Graphical user interface, text, application, email

Description automatically generated

A picture containing text

Description automatically generated

You can also use it for regular expressions

Text

Description automatically generated with low confidence

### Tapply()

Tapply(<what you’re applying the function to>, <how to categorize the returned values>,<method to apply>)

Text

Description automatically generated with low confidence

tapply() is used to apply a function over subsets of a vector. It is primarily used when we have the following circumstances:

1. A dataset that can be broken up into groups (via categorical variables - aka factors)
2. We desire to break the dataset up into groups
3. Within each group, we want to apply a function

The arguments to tapply() are as follows:

* x is a vector
* INDEX is a factor or a list of factors (or else they are coerced to factors)
* FUN is a function to be applied
* ... contains other arguments to be passed FUN
* simplify, should we simplify the result?

# syntax of tapply function

tapply(x, INDEX, FUN, ..., simplify = TRUE)

Text

Description automatically generated with low confidence

### Sapply()

The sapply function in R applies a function to a vector or list and returns a vector, a matrix or an array.

**sapply**(X, *# Vector, list or expression object*

FUN, *# Function to be applied*

..., *# Additional arguments to be passed to FUN*

simplify = TRUE, *# If FALSE returns a list. If "array" returns an array if possible*

USE.NAMES = TRUE) *# If TRUE and if X is a character vector, uses the names of X*

Graphical user interface, text, application

Description automatically generated

### str\_replace()

str\_replace(string, pattern, replacement)

The **str\_replace()** function from the [stringr](https://stringr.tidyverse.org/" \t "_blank) package in R can be used to replace matched patterns in a string. This function uses the following syntax:

**str\_replace(string, pattern, replacement)**

where:

* **string:** Character vector
* **pattern:** Pattern to look for
* **replacement:** A character vector of replacements

### str\_replace\_all()

str\_replace\_all(string, pattern, replacement)

### GetUrl()

library(httr)

AccidentURL <- "http://data.maryland.gov/api/views/pdvh-tf2u/rows.json?accessType=DOWNLOAD"

apiResult <- GET(AccidentURL)

## Accessing Data SQLDF readXLS

### Sqldf()

SQL select on data frames

sqldf("select \* from iris limit 5")

sqldf("select count(\*) from iris")

sqldf("select Species, count(\*) from iris group by Species")

# create a data frame

DF <- data.frame(a = 1:5, b = letters[1:5])

sqldf("select \* from DF")

sqldf("select avg(a) mean, variance(a) var from DF") # see example 15

### Read.xlsx()

### Dbconnect()

Connect to a database management system

con <- dbConnect(odbc::odbc(),

driver = "NetezzaSQL",

database = "SYSTEM",

uid = Sys.getenv("netezza\_username"),

pwd = Sys.getenv("netezza\_password"),

server = "NETEZZA\_SERVER",

port = 5480)

## Accessing Data Concepts

### Importing spreadsheets

### Accessing a database R & SQL

### JSON, why would we use JSON? What is the purpose?

We use JSON because it's extremely lightweight to send back and forth in HTTP requests and responses due to the small file size.

It's easy to read compared to something like XML since it's much cleaner and there's not as many opening and closing tags to worry about.

JSON, a data interchange format native to Javascript, is easier to deal with than the XML in the AJAX applications found in web clients.

### How and why would we use Tapply or Sapply?

**sapply()** function takes list, vector or data frame as input and gives output in vector or matrix. It is useful **for operations on list objects and returns a list object of same length of original set**. Sapply function in R does the same job as lapply() function but returns a vector.

**tapply() computes a measure (mean, median, min, max, etc..) or a function for each factor variable** in a vector. It is a very useful function that lets you create a subset of a vector and then apply some functions to each of the subset.

### Why do we need to remove NAs? What problems arise if we leave NAs?

Na’s can affect statistical calculations; certain procedures don’t handle missing data gracefully.

Removal of missing values can distort a regression analysis. This is particularly true if you are working with higher order or more complicated models.

## Plotting/Visualization

### Box Plot



Chart, box and whisker chart

Description automatically generated



Chart, box and whisker chart

Description automatically generated

## Concepts Plotting/Visualization

### How does ggplot work?

1. Source of data, map the data to the aesthetic of the graph
2. Add a geometric object, dots, line, histogram, heatmap etc

ggplot(data\_source, aes(*map data components to graph components*)) +

geom\_xxx(*arguments to modify the appearance of the geom*) +

theme\_xxx(*arguments to change the overall appearance*) +

labs(*add axis-labels and a title*)

* Starts with function ggplot()
  + Two args, source of data, second maps data components to the graph
* The second argument is the aes() – Aesthetic mapping, each argument within this is called an aesthetic
* Histogram example
  + Ggplot(airquality, aes(x=Temp))
* Then you must attach a geometric objects **geom()**
  + ggplot(airquality, aes(x=Temp)) + geom\_histomgram()

### What is a heat map?

**A heatmap is a graphical representation where individual values of a matrix are represented as colors.** A heatmap is very useful in visualizing the concentration of values between two dimensions of a matrix. This helps in finding patterns and gives a perspective of depth.

### Why are visualizations important?

**Data visualization helps to tell stories by curating data into a form easier to understand, highlighting the trends and outliers.** A good visualization tells a story, removing the noise from data and highlighting the useful information. ... Effective data visualization is a delicate balancing act between form and function.

### Histograms

Useful in gaining understanding of how data is distributed, is it a normal distribution? Is it skewed?

hist(attribute)} is data skewed, normal distribution

### When would we use a histogram rather than a scatter plot?

**Scatter plots are used when there is a real or implied continuity to the X variable data.**

### What is the difference between bins and binWidth?

Bins – number of bins or x axis containers for data within a histogram, bins are sometimes referred to as intervals classes or buckets.

BinWidth – the size of each bin, not defining the number of bins but what they represent.

### Why are bins useful?

Bins are useful for grouping data, defining a groups and or simplifying/defining the granularity of a visualization/plot.

### How does visualization compare with statistical analysis? Characteristics of effective visualization?

A good visualization should establish two aspects of the data being presented: **show connections within the data that are too complex to explain with words. Make it easier for the audience to quickly understand** the information presented **and consider the outcomes of that data.**

* Visually appealing
* Scalable
* Provides the needed information
* Accessible, modifiable
* Rapid development and deployment

### Why do we have a boxplot?

It **divides the data set into three quartiles**. This graph represents the minimum, maximum, median, first quartile and third quartile in the data set**. It is also useful in comparing the distribution of data across data sets by drawing boxplots for each of them**.

# Ggplot geom Coding

### geom\_boxplot()

The boxplot compactly displays the distribution of a continuous variable. It visualises five summary statistics (the median, two hinges and two whiskers), and all "outlying" points individually.

### geom\_hist()

Visualise the distribution of a single continuous variable by dividing the x axis into bins and counting the number of observations in each bin. Histograms (geom\_histogram()) display the counts with bars; frequency polygons (geom\_freqpoly()) display the counts with lines. Frequency polygons are more suitable when you want to compare the distribution across the levels of a categorical variable.

### ggtitle()

Add the title of the ggplot

### geom\_line()

connects them in order of the variable on the x axis.

### geom\_col()

### geom\_point()

draws points on the map

### geom\_text()

### coord\_flip()

flips the x and y coordinates

### theme()

### format()

### barplot()

# Creating a Function

### What is a function?

A function is a set of statements organized together to perform a specific task.

In R, a function is an object so the R interpreter is able to pass control to the function, along with arguments that may be necessary for the function to accomplish the actions.

The function in turn performs its task and returns control to the interpreter as well as any result which may be stored in other objects.

### Why do we use Functions?

We use functions to both optimize efficiency of the user and programming. For repetitive tasks it provides a quick code that we can execute with specific parameters/arguments either with an expected outcome or with the expectation that a value of some sort will be returned.

### How to create a function

A picture containing graphical user interface

Description automatically generated