SESSION II : R SOFTWARE

	Big Data	Small Data
Data Condition	Always unstructured, not ready for analysis, many relational database tables that need merged	Ready for analysis, flat file, no need for merging tables.
Location	Cloud, Offshore, SQL Server, etc.	Database, local PC
Data Size	Over 50K Variables, over 50K individuals, random samples, unstructured	File that is in a spreadsheet, that can be viewed on a few sheets of paper
Data Purpose	No intended purpose	Intended purpose for Data Collection

R Data Types

- What is R Data Types?

  It can handle complex statistical operations in an easy and optimized way.

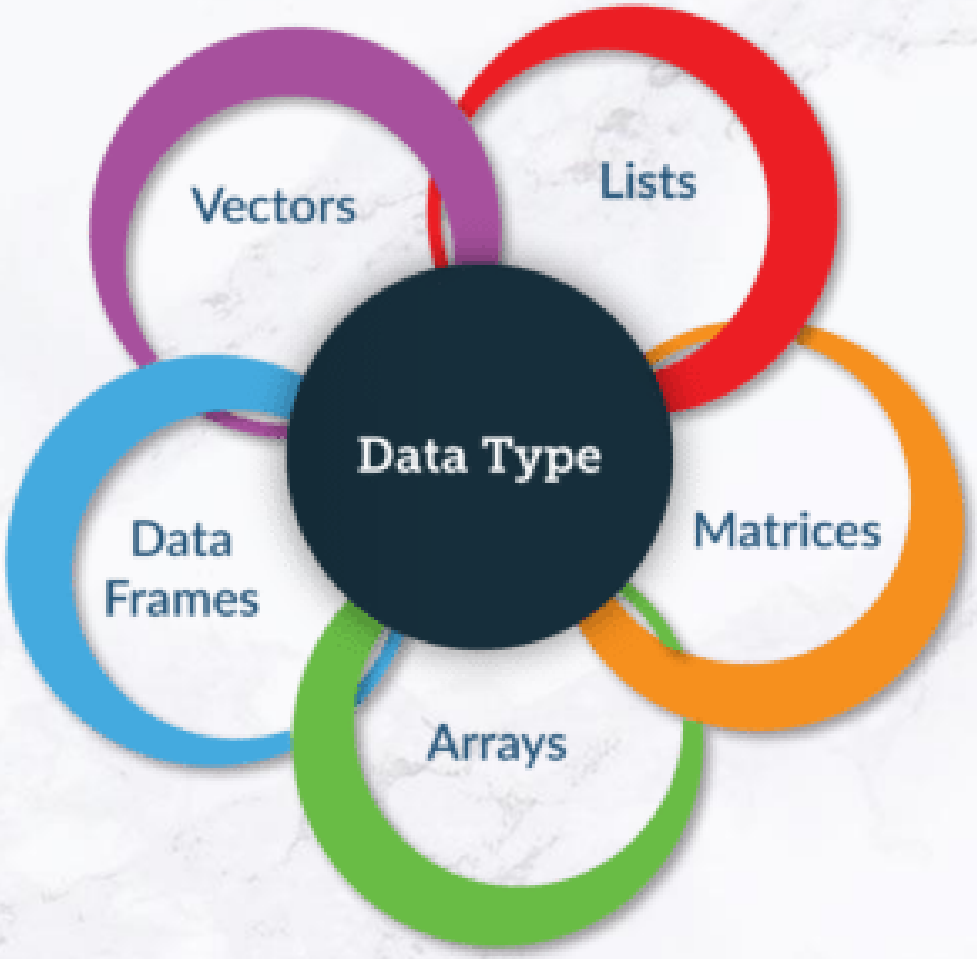
- **Vector** – A basic data structure of R containing the same type of data.

- **Lists** – Lists store collections of objects when vectors are of same type and length in a matrix.

- **Matrices** – A matrix is a rectangular array of numbers or other mathematical objects. We can do operations such as addition and multiplication on Matrix in R.

- **Arrays** – Arrays are the R data objects which can store data in more than two dimensions.

- **Data Frames** – Generated by combining together multiple vectors such that each vector becomes a separate column.



Basic Data Types

- **Logical** – TRUE / FALSE
- **Numeric** – 13, 3.5
- **Integer** – 25L, 40L
- **Complex** – 3+**2i**
- **Character** – “A” , ‘Hello World’
- **Raw** – “Hello” : 48 65 6c 6c 65

Variables

- A variable provides us with named storage that our programs can manipulate.

- A variable in R can store an atomic vector, group of atomic vectors or a combination of many R-Objects.

- A valid variable name consists of letters, numbers and the dot or underline characters. The variable name starts with a letter or the dot not followed by a number.

Variable Name	Validity	Reason
var_name2.	Valid	Has letters, numbers, dot and underscore.
var_name%	Invalid	Has the character '%'. Only dot(.) and underscore allowed.
2var_name	Invalid	Starts with a number
.var_name , var.name	Valid	Can start with a dot(.) but the dot(.)should not be followed by a number.
.2var_name	Invalid	The starting dot is followed by a number making it invalid.

Variables Assignment

Try:

```
var = 19
print(var)
```

Also try: print(“var”)

```
var ->  25
print(var)

25 <- var
print(var)
```

typeof()
class()

String Data:	x = “NMIMS” print(x)
Boolean Data:	y = TRUE print(y)
Integer Data:	z = 20L print(z)
Complex Data:	c = 2+5i print(c)
Raw Data:	t = charToRaw(“Hello”) print(t)
Numeric Data:	n = 1995 print(n)

Reading Input from user

```
x = readline()
print(x)
```

```
y = readline(prompt="Enter an Integer: ")
print(y)
```

Operators

- An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations.

- R language is rich in built-in operators and provides following types of operators.

- Types of Operators:
  1. Arithmetic Operators
  2. Relational Operators
  3. Logical Operators
  4. Assignment Operators
  5. Miscellaneous Operators

Arithmetic Operators

Operator	Description	Example
+	Adds two vectors	v <- c(2,5.5,6) t <- c(8, 3, 4) print(v+t)
-	Subtracts second vector from the first	print(v-t)
*	Multiplies both vectors	print(v*t)
/	Divide the first vector with the second	print(v/t)
%%	Give the remainder of the first vector with the second	print(v%%t)
%/%	The result of division of first vector with second (quotient)	print(v%/%t)
^	The first vector raised to the exponent of second vector	print(v^t)

Arithmetic Operators

```
a = as.integer(readline(prompt = “Enter First No. : “))
print(a); typeof(a)

b = as.integer(readline(prompt = “Enter Second No: “))
print(b); typeof(b)

c = a+b
print(“Sum”)
print(c)

print(paste(“Sum: “, a+b))
print(paste(“Sub: “, a-b))
print(paste(“Mul: “, a*b))
print(paste(“Exp: “, a^b))
print(paste(“Mod: “, a%%b))
print(paste(“IntDiv: “, a%/%b))
```

Relational Operators

Operator	Description	Example
>	Checks if each element of the first vector is greater than the corresponding element of the second vector.	v <- c(2,5.5,6,9) t <- c(8,2.5,14,9) print(v>t)
<	Checks if each element of the first vector is less than the corresponding element of the second vector.	print(v < t)
==	Checks if each element of the first vector is equal to the corresponding element of the second vector.	print(v == t)
<=	Checks if each element of the first vector is less than or equal to the corresponding element of the second vector.	print(v<=t)
>=	Checks if each element of the first vector is greater than or equal to the corresponding element of the second vector.	print(v>=t)
!=	Checks if each element of the first vector is unequal to the corresponding element of the second vector.	print(v!=t)

Relational Operators

```
x = as.integer(readline(prompt = “Enter First No. : “))
y = as.integer(readline(prompt = “Enter Second No: “))


z = x<y ; print(z)
z1 = x>y: print(z1)
```

Logical Operators

Operator	Description	Example
&	It is called Element-wise Logical AND operator. It combines each element of the first vector with the corresponding element of the second vector and gives a output TRUE if both the elements are TRUE.	v <- c(3,1,TRUE,2+3i) t<-c(4,1,FALSE,2+3i) print(v&t)
|	It is called Element-wise Logical OR operator. It combines each element of the first vector with the corresponding element of the second vector and gives a output TRUE if one the elements is TRUE.	v <- c(3,0,TRUE,2+2i) t <- c(4,0,FALSE,2+3i) print(v|t)
!	It is called Logical NOT operator. Takes each element of the vector and gives the opposite logical value.	v <- c(3,0,TRUE,2+2i) print(!v)

Operator	Description	Example
&&	Called Logical AND operator. Takes first element of both the vectors and gives the TRUE only if both are TRUE.	v <- c(3,0,TRUE,2+2i) t <- c(1,3,TRUE,2+3i) print(v&&t)
||	Called Logical OR operator. Takes first element of both the vectors and gives the TRUE if one of them is TRUE.	v <- c(0,0,TRUE,2+2i) t <- c(0,3,TRUE,2+3i) print(v||t)

Miscellaneous Operators

Operator	Description	Example
:	Colon operator. It creates the series of numbers in sequence for a vector.	v <- 2:8 print(v)
%in%	This operator is used to identify if an element belongs to a vector.	v1 <- 8 v2 <- 12 t <- 1:10 print(v1 %in% t) print(v2 %in% t)
%*%	This operator is used to multiply a matrix with its transpose.	M = matrix(c(2,6,5,1,10,4), nrow = 2,ncol = 3,byrow = TRUE) t = M %*% t(M) print(t)

Vectors

```
data = c(10,20,30,40,50)

print(data)
print(data[3])
```

Miscellaneous Operator:

1. Colon Operator

```
v = 2:8
print(v)
```

2. Membership Operator

```
v1 = 9
v2 = 12
t = 1:10
print(t)
print(v1 %in% t)
```

Decision Making Statements

There are the following variants of if statement in R language.

- ❑ If Statement
- ❑ If-else Statement
- ❑ Multiple if Statement
- ❑ If else-if ladder
- ❑ Nested if statement
- ❑ Switch statement

We have following conditional statements:

- Use **if** to specify a block of code to be executed, if a specified condition is true.
- Use **else** to specify a block of code to be executed, if the same condition is false.
- Use **else if** to specify a new condition to test, if the first condition is false.
- Use **switch** to select one of many blocks of code to be executed.