R Notebook

http://rmarkdown.rstudio.com/r_notebooks.html

This is an R Markdown Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Ctrl+Shift+Enter*.

```
d1 <- 2+3
d1
```

```
## [1] 5
```

```
array_1 <-array(1:24, dim=c(3,4,2))
array_1
```

```
## , , 1
##
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
##
## , , 2
##
##      [,1] [,2] [,3] [,4]
## [1,]   13   16   19   22
## [2,]   14   17   20   23
## [3,]   15   18   21   24
```

Creating our very First Array

```
?array()

array_1 <-array(1:24, dim=c(3,4,3))
array_1
```

```
## , , 1
##
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
##
## , , 2
##
##      [,1] [,2] [,3] [,4]
## [1,]   13   16   19   22
## [2,]   14   17   20   23
## [3,]   15   18   21   24
##
## , , 3
##
##      [,1] [,2] [,3] [,4]
```

```
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

DIM == "Integer vector of length one or more giving the maximal indices in each dimension." Above we have ROWS or INDEX == 3 , COLUMNS == 4 and DIMENSIONS == 3 VALUES or OBSERVATIONS are filled in COLUMNS FIRST - ROWS NEXT

```
is.array(array_1)
```

```
## [1] TRUE
```

```
array_2 <-array(1:24, dim=c(3,4))
array_2
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

```
typeof(array_2)
```

```
## [1] "integer"
```

```
typeof(array_1)
```

```
## [1] "integer"
```

Both ARRAYS Stored as INTEGERS in Memory

WHAT ARE - VECTORS — One Dimension Arrays which can hold - NUMERIC , CHARACTER or LOGICAL DATA
Vectors can be of THREE Types or MODES - which means the DATA TYPE they can hold.
#
A VECTOR is created using the - COMBINE Function()
#
Lets see an example of each type below -

```
num_vector <- c(22,22,33,33,44)
num_vector
```

```
## [1] 22 22 33 33 44
```

```
char_vector <- c("x","d","c","f")
char_vector
```

```
## [1] "x" "d" "c" "f"
```

```
logical_vector <- c(TRUE,FALSE,TRUE,FALSE,FALSE,FALSE)
logical_vector
```

```
## [1]  TRUE FALSE  TRUE FALSE FALSE FALSE
```

SCALARS - One Element Vectors - useful for holding CONSTANT values

```
a1 <- "FINANCE"
b1 <- "MARKETING"
c1 <- "SALES"
d1 <- 3.1416
a1
```

```
## [1] "FINANCE"
```

```
b1
```

```
## [1] "MARKETING"
```

```
c1
```

```
## [1] "SALES"
```

```
d1
```

```
## [1] 3.1416
```

We refer ELEMENTS of a VECTOR by mentioning their INDEX - STARTING at 1

Take an example of the "num_vector" from above , we REFER the 1st ELEMENT - Element 1 as below , assign it to a VAR "nm_1" and then print the value stored within the VAR :-

```
nm_1 <- num_vector[1]
nm_1
```

```
## [1] 22
```

Below we refer value stored in Elements - 2,3 and 5 , the values as seen below are - 22, 33 and 44

```
nm_2 <- num_vector[2]
nm_3 <- num_vector[3]
nm_5 <- num_vector[5]

nm_2
```

```
## [1] 22
```

```
nm_3
```

```
## [1] 33
```

```
nm_5
```

```
## [1] 44
```

We can also access a RANGE of elements by refering the INDEX of START and STOP Elements :-

```
nm_range <- num_vector[2:5]
nm_range
```

```
## [1] 22 33 33 44
```

MATRICES (Plural) MATRIX (Singular) - again like a VECTOR can store values in elements of the SAME MODE or SAME TYPE .

Matrices are TWO DIMENSIONAL ARRAYS of Data .

```
?matrix()
```

matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)

```
m_1 <- matrix(data=66:99,nrow=2,ncol=2)
m_1
```

```
##      [,1] [,2]
## [1,]   66   68
## [2,]   67   69
```

Data set has been truncated as we didnt mention enough ROWS for the MATRIX , lets try again with 5 ROWS ....

```
m_1 <- matrix(data=66:99,nrow=5,ncol=2)
```

```
## Warning in matrix(data = 66:99, nrow = 5, ncol = 2): data length [34] is
## not a sub-multiple or multiple of the number of rows [5]
```

```
m_1
```

```
##      [,1] [,2]
## [1,]   66   71
## [2,]   67   72
## [3,]   68   73
## [4,]   69   74
## [5,]   70   75
```

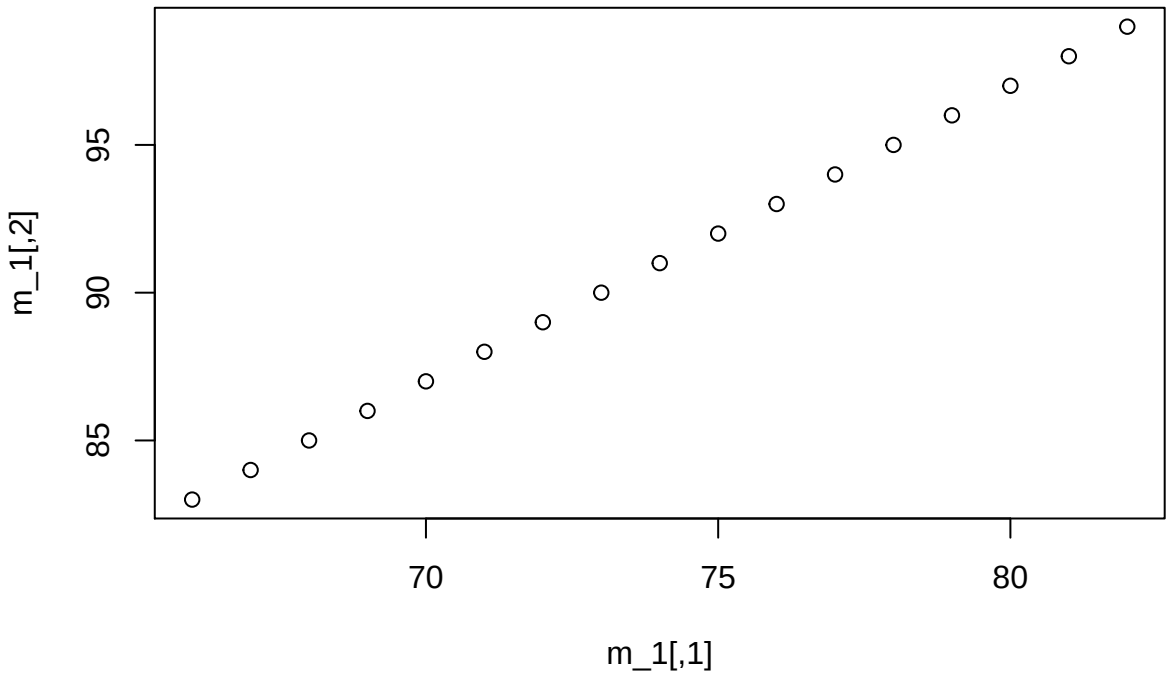
For Data Length 34 (data being numeric values from 66 to 99) , we need to provide a "multiple" or "sub-multiple"of 34 as the ROWS value - lets try with 17.

```
m_1 <- matrix(data=66:99,nrow=17,ncol=2)
m_1
```

```
##       [,1] [,2]
##  [1,]   66   83
##  [2,]   67   84
##  [3,]   68   85
##  [4,]   69   86
##  [5,]   70   87
##  [6,]   71   88
##  [7,]   72   89
##  [8,]   73   90
##  [9,]   74   91
## [10,]   75   92
## [11,]   76   93
## [12,]   77   94
## [13,]   78   95
## [14,]   79   96
## [15,]   80   97
## [16,]   81   98
## [17,]   82   99
```

As seen above now the MATRIX - m_1 , can fit in the Data Values just fine with - 17 ROWS and 2 COLUMNS.

```
plot(m_1)
```



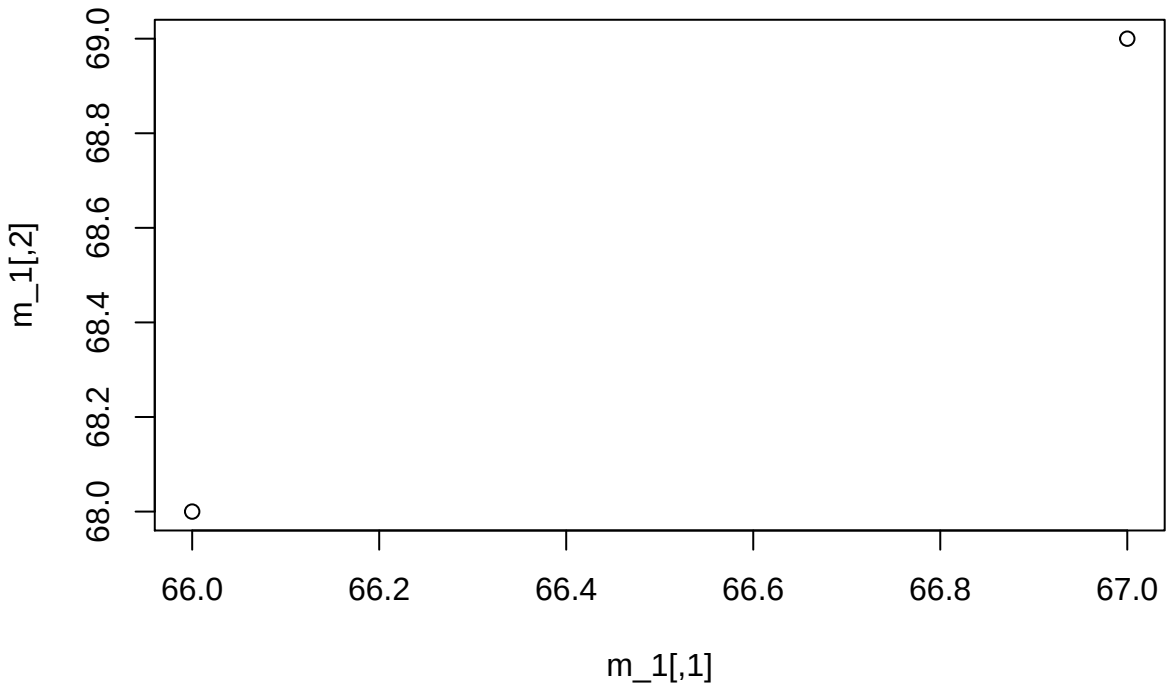
We can also reverse the ORDER of COLUMNS and ROWS

```
m_1 <- matrix(data=66:99,nrow=2,ncol=17)
m_1
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
## [1,]   66   68   70   72   74   76   78   80   82    84    86    88    90
## [2,]   67   69   71   73   75   77   79   81   83    85    87    89    91
##      [,14] [,15] [,16] [,17]
## [1,]    92    94    96    98
## [2,]    93    95    97    99
```

We still derive a MATRIX - m_1 , but the plot as seen below is illegible.

```
plot(m_1)
```



Weve seen the basic arguments for the Function "matrix()" above - now we look at "byrow" and "dimnames". DimNames or Dimension Names can be a LIST of structure - LIST(ROW_NAMES,COLUMN_NAMES) . If we have only the 1st ELEMENT of a LIST then that defaults to ROW_NAMES and if we have a 2nd ELEMENT it defaults to COLUMN_NAMES.
Seen below - row_names and col_names created using the COMBINE - c() , function.

```
row_names <- c("Row_Name_1","Row_Name_2")
col_names <- c("Col_Name_1","Col_Name_2")
m_1 <- matrix(data=66:69,nrow=2,ncol=2,byrow=FALSE,dimnames=list(col_names,row_names))
m_1
```

```
##            Row_Name_1 Row_Name_2
## Col_Name_1         66         68
## Col_Name_2         67         69
```

As seen above when we dont specify ROW and COLUMN names we see a DIFFERENT Notation for default Columns and Rows Index values.

```
m_1 <- matrix(data=66:69,nrow=2,ncol=2)
m_1
```

```
##      [,1] [,2]
## [1,]   66   68
## [2,]   67   69
```

The ROWS are represented by INDEX of the kind – [1,] [2,]

The COLUMNS with – [,1] [,2]

Thus the MATRIX values, can also be accessed in this manner as seen below ...

```
m_1[1,1]
```

```
## [1] 66
```

```
#
m_1[1,2]
```

```
## [1] 68
```

```
#
m_1[2,1]
```

```
## [1] 67
```

```
#
m_1[2,2]
```

```
## [1] 69
```

Weve seen examples of VECTORS and MATRICES in the above sections .

Both MATRICES and VECTORS can store data of only 1 Mode or TYPE and are Both 2D or TWO DIMENSIONAL in shape .

We have also seen ARRAYS which are again SINGLE MODE or TYPE but 3D or THREE DIMENSIONAL in shape .

In the section below we look at DATAFRAMES - which we probably end up using the most in our day to day analysis as they can hold MULTIPLE TYPES of DATA or are MULTIMODE in structure.

We shall import a .CSV File to begin with and create a DATAFRAME from the same

We also at a later stage look at creating own DataFrames from Manual inputs within R

?read.csv()

read.csv(file, header = TRUE, sep = ",", quote = "\"", dec = ".", fill = TRUE, comment.char = "", ...)

```
df_1 <- read.csv("~/Desktop/R_Own/R_1 - Sheet1.csv",header =TRUE , sep = "," )
df_1
```

```
##    X.    Product.Name        Prod.ID Date.of.Invoice Date.of.Shipping
## 1   1 OFF-LA-10002782 MX-2014-143658      01-01-2013       02-01-2013
## 2   2 FUR-FU-10004015 MX-2012-155047      01-01-2013       02-01-2013
## 3   3 FUR-BO-10002352 MX-2012-155047      01-01-2013       02-01-2013
## 4   4 OFF-BI-10004428 MX-2012-155047      01-01-2013       02-01-2013
## 5   5 OFF-AR-10004594 MX-2012-155047      01-01-2013       02-01-2013
## 6   6 OFF-EN-10001375 MX-2012-155047      01-01-2013       02-01-2013
## 7   7 OFF-EN-10001375 MX-2013-134096      01-01-2013       02-01-2013
## 8   8 TEC-MA-10004956 MX-2013-134096      01-01-2013       02-01-2013
## 9   9 OFF-SU-10003474 MX-2013-134096      01-01-2013       02-01-2013
## 10 10 TEC-AC-10001830 MX-2013-134096      01-01-2013       02-01-2013
## 11 11 OFF-BI-10002075 MX-2013-134096      01-01-2013       02-01-2013
## 12 12 OFF-FA-10002526 MX-2013-156335      01-01-2013       02-01-2013
## 13 13 FUR-CH-10002846 MX-2013-156335      01-01-2013       02-01-2013
## 14 14 OFF-EN-10004100 MX-2014-121923      02-01-2013       04-01-2013
## 15 15 OFF-AR-10003914 MX-2014-135706      02-01-2013       03-01-2013
## 16 16 OFF-FA-10000038 MX-2014-135706      02-01-2013       03-01-2013
## 17 17 OFF-EN-10000761 US-2013-126655      02-01-2013       03-01-2013
## 18 18 FUR-FU-10003066 US-2013-126655      02-01-2013       03-01-2013
## 19 19 OFF-EN-10000075 US-2013-126655      02-01-2013       03-01-2013
```

```
## 20 20 OFF-EN-10002226 US-2013-126655      02-01-2013       03-01-2013
## 21 21 FUR-CH-10002132 MX-2013-167759      02-01-2013       04-01-2013
## 22 22 TEC-AC-10002749 MX-2013-163139      02-01-2013       02-01-2013
## 23 23 OFF-SU-10000066 MX-2013-163139      02-01-2013       02-01-2013
## 24 24 OFF-BI-10003934 US-2014-119753      02-01-2013       03-01-2013
## 25 25 OFF-BI-10003932 US-2012-133970      02-01-2013       03-01-2013
##    Cost.Price Quantity Sales.Price Shipping.Index Shipping.Type
## 1      13.080        3      39.240              1      PRIORITY
## 2     252.160        8    2017.280              2      PRIORITY
## 3     193.280        2     386.560              3      PRIORITY
## 4      35.440        4     141.760              4      PRIORITY
## 5      71.600        2     143.200              5      PRIORITY
## 6      56.120        2     112.240              6      PRIORITY
## 7      56.120        2     112.240              7      STANDARD
## 8     344.640        3    1033.920              8      STANDARD
## 9      97.360        4     389.440              9      STANDARD
## 10    341.520        2     683.040             10      STANDARD
## 11     12.060        3      36.180             11      STANDARD
## 12     20.760        3      62.280             12      STANDARD
## 13    210.640        4     842.560             13      STANDARD
## 14     80.100        3     240.300             14      STANDARD
## 15    132.640        4     530.560             15      STANDARD
## 16     12.940        1      12.940             16      STANDARD
## 17     18.840        2      37.280             17      STANDARD
## 18    308.280        7    2157.560             18      STANDARD
## 19     40.176        2      79.952             19      STANDARD
## 20      8.784        3      25.952             20      PRIORITY
## 21    273.472        4    1093.688             21      PRIORITY
## 22     27.000        1      27.000             22      PRIORITY
## 23    207.000        9    1863.000             23      PRIORITY
## 24     60.660        3     181.580             24      PRIORITY
## 25    181.116        9    1629.644             25      PRIORITY
##           Category
## 1  Office Supplies
## 2        Furniture
## 3        Furniture
## 4  Office Supplies
## 5  Office Supplies
## 6  Office Supplies
## 7  Office Supplies
## 8       Technology
## 9  Office Supplies
## 10      Technology
## 11 Office Supplies
## 12 Office Supplies
## 13       Furniture
## 14 Office Supplies
## 15 Office Supplies
## 16 Office Supplies
## 17 Office Supplies
## 18       Furniture
## 19 Office Supplies
## 20 Office Supplies
## 21       Furniture
```

```
## 22      Technology
## 23 Office Supplies
## 24 Office Supplies
## 25 Office Supplies
```

As seen above there are various ways of reading in the Data from the .CSV format .

We used the read.csv() function which is present within the {utils} inbuilt package and is shipped along with - read.table() , read.csv2() , read.delim() and read.delim2()

We shall mostly be using - read.csv() and read.table()

Also as seen above – within the display of df_1 - the column names have the TYPE or MODE mentioned below - the "fctr" is the FACTOR , the "int" being the INTEGER etc .

Also this can be seen as such within the RIGHT PANEL of RSTudio in the ENVIRONEMENT >> Global Environment - section.

Kindly note that SCREENSHOTS of these can be seen within the GITHUB Repository Read Me file and path- "https://github.com/RohitDhankar/R-Beginners-Online-Virtual-Learning-Session/tree/master/Images_R_ScreenShots"

We need not mention the DEFAULTS and as seen below read.csv() - would work as well with just - the file path of the file.

```
df_2 <- read.csv("~/Desktop/R_Own/R_1 - Sheet1.csv")
df_2
```

```
##    X.    Product.Name        Prod.ID Date.of.Invoice Date.of.Shipping
## 1   1 OFF-LA-10002782 MX-2014-143658      01-01-2013       02-01-2013
## 2   2 FUR-FU-10004015 MX-2012-155047      01-01-2013       02-01-2013
## 3   3 FUR-BO-10002352 MX-2012-155047      01-01-2013       02-01-2013
## 4   4 OFF-BI-10004428 MX-2012-155047      01-01-2013       02-01-2013
## 5   5 OFF-AR-10004594 MX-2012-155047      01-01-2013       02-01-2013
## 6   6 OFF-EN-10001375 MX-2012-155047      01-01-2013       02-01-2013
## 7   7 OFF-EN-10001375 MX-2013-134096      01-01-2013       02-01-2013
## 8   8 TEC-MA-10004956 MX-2013-134096      01-01-2013       02-01-2013
## 9   9 OFF-SU-10003474 MX-2013-134096      01-01-2013       02-01-2013
## 10 10 TEC-AC-10001830 MX-2013-134096      01-01-2013       02-01-2013
## 11 11 OFF-BI-10002075 MX-2013-134096      01-01-2013       02-01-2013
## 12 12 OFF-FA-10002526 MX-2013-156335      01-01-2013       02-01-2013
## 13 13 FUR-CH-10002846 MX-2013-156335      01-01-2013       02-01-2013
## 14 14 OFF-EN-10004100 MX-2014-121923      02-01-2013       04-01-2013
## 15 15 OFF-AR-10003914 MX-2014-135706      02-01-2013       03-01-2013
## 16 16 OFF-FA-10000038 MX-2014-135706      02-01-2013       03-01-2013
## 17 17 OFF-EN-10000761 US-2013-126655      02-01-2013       03-01-2013
## 18 18 FUR-FU-10003066 US-2013-126655      02-01-2013       03-01-2013
## 19 19 OFF-EN-10000075 US-2013-126655      02-01-2013       03-01-2013
## 20 20 OFF-EN-10002226 US-2013-126655      02-01-2013       03-01-2013
## 21 21 FUR-CH-10002132 MX-2013-167759      02-01-2013       04-01-2013
## 22 22 TEC-AC-10002749 MX-2013-163139      02-01-2013       02-01-2013
## 23 23 OFF-SU-10000066 MX-2013-163139      02-01-2013       02-01-2013
## 24 24 OFF-BI-10003934 US-2014-119753      02-01-2013       03-01-2013
## 25 25 OFF-BI-10003932 US-2012-133970      02-01-2013       03-01-2013
```

```
##    Cost.Price Quantity Sales.Price Shipping.Index Shipping.Type
## 1      13.080        3      39.240              1      PRIORITY
## 2     252.160        8    2017.280              2      PRIORITY
## 3     193.280        2     386.560              3      PRIORITY
## 4      35.440        4     141.760              4      PRIORITY
## 5      71.600        2     143.200              5      PRIORITY
## 6      56.120        2     112.240              6      PRIORITY
## 7      56.120        2     112.240              7      STANDARD
## 8     344.640        3    1033.920              8      STANDARD
## 9      97.360        4     389.440              9      STANDARD
## 10    341.520        2     683.040             10      STANDARD
## 11     12.060        3      36.180             11      STANDARD
## 12     20.760        3      62.280             12      STANDARD
## 13    210.640        4     842.560             13      STANDARD
## 14     80.100        3     240.300             14      STANDARD
## 15    132.640        4     530.560             15      STANDARD
## 16     12.940        1      12.940             16      STANDARD
## 17     18.840        2      37.280             17      STANDARD
## 18    308.280        7    2157.560             18      STANDARD
## 19     40.176        2      79.952             19      STANDARD
## 20      8.784        3      25.952             20      PRIORITY
## 21    273.472        4    1093.688             21      PRIORITY
## 22     27.000        1      27.000             22      PRIORITY
## 23    207.000        9    1863.000             23      PRIORITY
## 24     60.660        3     181.580             24      PRIORITY
## 25    181.116        9    1629.644             25      PRIORITY
##           Category
## 1  Office Supplies
## 2        Furniture
## 3        Furniture
## 4  Office Supplies
## 5  Office Supplies
## 6  Office Supplies
## 7  Office Supplies
## 8       Technology
## 9  Office Supplies
## 10      Technology
## 11 Office Supplies
## 12 Office Supplies
## 13       Furniture
## 14 Office Supplies
## 15 Office Supplies
## 16 Office Supplies
## 17 Office Supplies
## 18       Furniture
## 19 Office Supplies
## 20 Office Supplies
## 21       Furniture
## 22      Technology
## 23 Office Supplies
## 24 Office Supplies
## 25 Office Supplies
```

To refer to the elements of a DATAFRAME we have the following options :-

```r
df_2[1,3]
```

```
## [1] MX-2014-143658
## 11 Levels: MX-2012-155047 MX-2013-134096 MX-2013-156335 ... US-2014-119753
```

```r
#
df_2[3,1]
```

```
## [1] 3
```

```r
#
df_2[4,3]
```

```
## [1] MX-2012-155047
## 11 Levels: MX-2012-155047 MX-2013-134096 MX-2013-156335 ... US-2014-119753
```

```r
# Date of Invoice
#
#df_2["Date of Invoice"]
#
df_2["Category"]
```

```
##           Category
## 1  Office Supplies
## 2        Furniture
## 3        Furniture
## 4  Office Supplies
## 5  Office Supplies
## 6  Office Supplies
## 7  Office Supplies
## 8       Technology
## 9  Office Supplies
## 10      Technology
## 11 Office Supplies
## 12 Office Supplies
## 13       Furniture
## 14 Office Supplies
## 15 Office Supplies
## 16 Office Supplies
## 17 Office Supplies
## 18       Furniture
## 19 Office Supplies
## 20 Office Supplies
## 21       Furniture
## 22      Technology
## 23 Office Supplies
## 24 Office Supplies
## 25 Office Supplies
```

```r
#
df_2["Date.of.Invoice"]
```

```
##    Date.of.Invoice
## 1       01-01-2013
## 2       01-01-2013
## 3       01-01-2013
## 4       01-01-2013
```

```
## 5       01-01-2013
## 6       01-01-2013
## 7       01-01-2013
## 8       01-01-2013
## 9       01-01-2013
## 10      01-01-2013
## 11      01-01-2013
## 12      01-01-2013
## 13      01-01-2013
## 14      02-01-2013
## 15      02-01-2013
## 16      02-01-2013
## 17      02-01-2013
## 18      02-01-2013
## 19      02-01-2013
## 20      02-01-2013
## 21      02-01-2013
## 22      02-01-2013
## 23      02-01-2013
## 24      02-01-2013
## 25      02-01-2013
```

```
# Note the SPACES bewteen words are denoted by PERIODS.
#
```

df_2[c("Category","Date.of.Invoice")] Two COLUMNS or VARIABLES from df_2 assigned to create df_3 , using the COMBINE function.

```
df_3 <- df_2[c("Category","Date.of.Invoice")]
df_3
```

```
##           Category Date.of.Invoice
## 1  Office Supplies      01-01-2013
## 2        Furniture      01-01-2013
## 3        Furniture      01-01-2013
## 4  Office Supplies      01-01-2013
## 5  Office Supplies      01-01-2013
## 6  Office Supplies      01-01-2013
## 7  Office Supplies      01-01-2013
## 8       Technology      01-01-2013
## 9  Office Supplies      01-01-2013
## 10      Technology      01-01-2013
## 11 Office Supplies      01-01-2013
## 12 Office Supplies      01-01-2013
## 13       Furniture      01-01-2013
## 14 Office Supplies      02-01-2013
## 15 Office Supplies      02-01-2013
## 16 Office Supplies      02-01-2013
## 17 Office Supplies      02-01-2013
## 18       Furniture      02-01-2013
## 19 Office Supplies      02-01-2013
## 20 Office Supplies      02-01-2013
```

```
## 21       Furniture      02-01-2013
## 22      Technology      02-01-2013
## 23 Office Supplies      02-01-2013
## 24 Office Supplies      02-01-2013
## 25 Office Supplies      02-01-2013
```

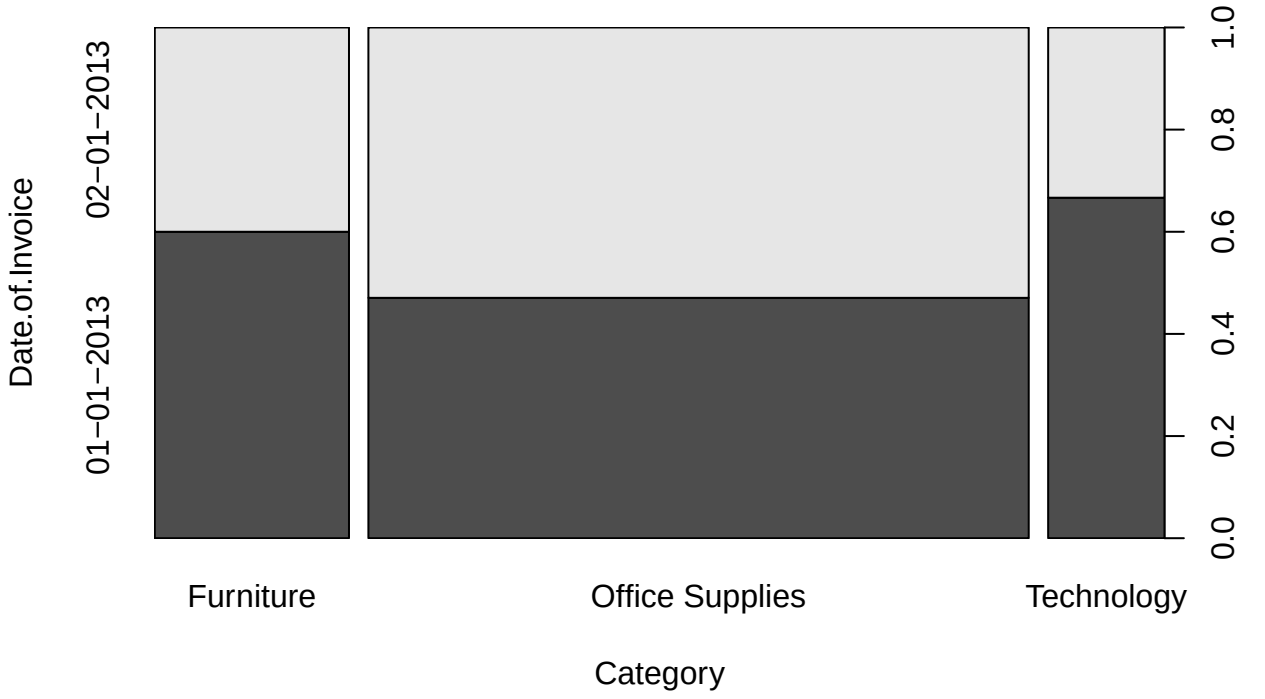
Refer any COLUMN or VARIABLE of a DF using the DOLLAR Notation .

```
df_4 <- df_3$Category
df_4
```

```
##  [1] Office Supplies Furniture       Furniture       Office Supplies
##  [5] Office Supplies Office Supplies Office Supplies Technology
##  [9] Office Supplies Technology      Office Supplies Office Supplies
## [13] Furniture       Office Supplies Office Supplies Office Supplies
## [17] Office Supplies Furniture       Office Supplies Office Supplies
## [21] Furniture       Technology      Office Supplies Office Supplies
## [25] Office Supplies
## Levels: Furniture Office Supplies Technology
```

A basic plot() of the DataFrame.

```
plot(df_3)
```



sessionInfo()

R version 3.3.2 (2016-10-31) Platform: x86_64-pc-linux-gnu (64-bit) Running under: Ubuntu 16.04.1 LTS

locale: [1] LC_CTYPE=en_IN.UTF-8 LC_NUMERIC=C LC_TIME=en_IN.UTF-8 LC_COLLATE=en_IN.UTF-8
[5] LC_MONETARY=en_IN.UTF-8 LC_MESSAGES=en_IN.UTF-8 LC_PAPER=en_IN.UTF-8 LC_NAME=C
[9] LC_ADDRESS=C LC_TELEPHONE=C LC_MEASUREMENT=en_IN.UTF-8 LC_IDENTIFICATION=C

attached base packages: [1] stats graphics grDevices utils datasets methods base

loaded via a namespace (and not attached): [1] backports_1.0.4 magrittr_1.5 rprojroot_1.1 htmltools_0.3.5 tools_3.3.2 base64enc_0.1-3 yaml_2.1.14
[8] Rcpp_0.12.8 stringi_1.1.2 rmarkdown_1.3 knitr_1.15.1 jsonlite_1.1 stringr_1.1.0 digest_0.6.10
[15] evaluate_0.10

EOF - R_1.Rmd