***Problem statement1. - Read multiple JSON files into a directory to convert into a dataset.***

***Solution: -***

JSON file stores data as text in readable format. JSON stands for “JavaScript Object Notation”. R can read JSON files using the rjson package. Issue the following command

# Load the package required to read JSON files.

> install.packages("rjson")

The JSON file is read by R using the function from JSON(). It is stored as list in R

**>**library("rjson")

# Give the input file name to the function.

>result<- fromJSON(file = "C:/Users/mpkee/OneDrive/Desktop/.JSON")

>print(result)

#above code will produce the following result

$ID

[1] "1" "2" "3" "4" "5" "6" "7" "8"

$Name

[1] "Rick" "Dan" "Michelle" "Ryan" "Gary"

[6] "Nina" "Simon" "Guru"

$Salary

[1] "623.3" "515.2" "611" "729" "843.25" "578"

[7] "632.8" "722.5"

$StartDate

[1] "1/1/2012" "9/23/2013" "11/15/2014" "5/11/2014"

[5] "3/27/2015" "5/21/2013" "7/30/2013" "6/17/2014"

$Dept

[1] "IT" "Operations" "IT" "HR"

[5] "Finance" "IT" "Operations" "Finance"

# Convert JSON to a Data Frame

# Load the package required to read JSON files.

>library("rjson")

# Give the input file name to the function.

>result<- fromJSON(file = "C:/Users/mpkee/OneDrive/Desktop/.JSON")

# Convert JSON file to a data frame.

>json\_data\_frame<- as.data.frame(result)

print(json\_data\_frame)

When we execute the above code, it produces the following result −

|  |
| --- |
| > print(json\_data\_frame)  ID Name Salary StartDate Dept  1 1 Rick 623.3 1/1/2012 IT  2 2 Dan 515.2 9/23/2013 Operations  3 3 Michelle 611 11/15/2014 IT  4 4 Ryan 729 5/11/2014 HR  5 5 Gary 843.25 3/27/2015 Finance  6 6 Nina 578 5/21/2013 IT  7 7 Simon 632.8 7/30/2013 Operations  8 8 Guru 722.5 6/17/2014 Finance |
|  |
| |  | | --- | | > | |

***Problem statement2: -*  Parse the following JSON into a data frame.**

**js<-'{"name": null, "release\_date\_local": null, "title": "3 (2011)",**

**"opening\_weekend\_take": 1234, "year": 2011,**

**"release\_date\_wide": "2011-09-16", "gross": 59954}'**

**Solution: - Read the JSON File**

>js<-'{"name": null, "release\_date\_local": null, "title": "3 (2011)","opening\_weekend\_take": 1234, "year": 2011,"release\_date\_wide": "2011-09-16", "gross": 59954}'

>js<- fromJSON(js)

> print(js)

It Produces the following result

|  |
| --- |
| > print(js)  $name  NULL  $release\_date\_local  NULL  $title  [1] "3 (2011)"  $opening\_weekend\_take  [1] 1234  $year  [1] 2011  $release\_date\_wide  [1] "2011-09-16"  $gross  [1] 59954 |
|  |
| |  | | --- | | > | |

# Convert JSON to a Data Frame

> data\_frame<- as.data.frame(js)

Error in (function (..., row.names = NULL, check.rows = FALSE, check.names = TRUE, :

arguments imply differing number of rows: 0, 1

> js<- lapply(js, function(x), {x[sapply(x, is.null)])<- NA, until(x)})

Error: unexpected ',' in "js<- lapply(js, function(x),"

> js<- lapply(js, function(x) {x[sapply(x, is.null)])<- NA, until(x)})

Error: unexpected ')' in "js<- lapply(js, function(x) {x[sapply(x, is.null)])"

> json\_dataframe<- as.data.frame(js)

Error in (function (..., row.names = NULL, check.rows = FALSE, check.names = TRUE, :

arguments imply differing number of rows: 0, 1

> js <- lapply(js, function(x) {x[sapply(x, is.null)] <- NA})

Error in x[sapply(x, is.null)] <- NA : invalid subscript type 'list'

> js <- lapply(js, function(x) {x[sapply(x, is.NULL)] <- NA})

C:\Users\mpkee\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\110D65CD.tmp Show Traceback

C:\Users\mpkee\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\C25F4663.tmp Rerun with Debug

Error in match.fun(FUN) : object 'is.NULL' not found

Unable to convert it into Data Fram

***Problem Statement3: -Write a script for Variable Binning using R.***

Say I have a Vector with 100 values. I just need to bin it into 25 equal intervels for which I would then have to calculate the median.

|  |
| --- |
| > v<-c(1:100)  > v  [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15  [16] 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30  [31] 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45  [46] 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60  [61] 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75  [76] 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90  [91] 91 92 93 94 95 96 97 98 99 100  > tapply(v, cut(v, 25), median)  (0.901,4.96] (4.96,8.92] (8.92,12.9] (12.9,16.8] (16.8,20.8]  2.5 6.5 10.5 14.5 18.5  (20.8,24.8] (24.8,28.7] (28.7,32.7] (32.7,36.6] (36.6,40.6]  22.5 26.5 30.5 34.5 38.5  (40.6,44.6] (44.6,48.5] (48.5,52.5] (52.5,56.4] (56.4,60.4]  42.5 46.5 50.5 54.5 58.5  (60.4,64.4] (64.4,68.3] (68.3,72.3] (72.3,76.2] (76.2,80.2]  62.5 66.5 70.5 74.5 78.5  (80.2,84.2] (84.2,88.1] (88.1,92.1] (92.1,96] (96,100]  82.5 86.5 90.5 94.5 98.5 |
|  |
| |  | | --- | | > | |