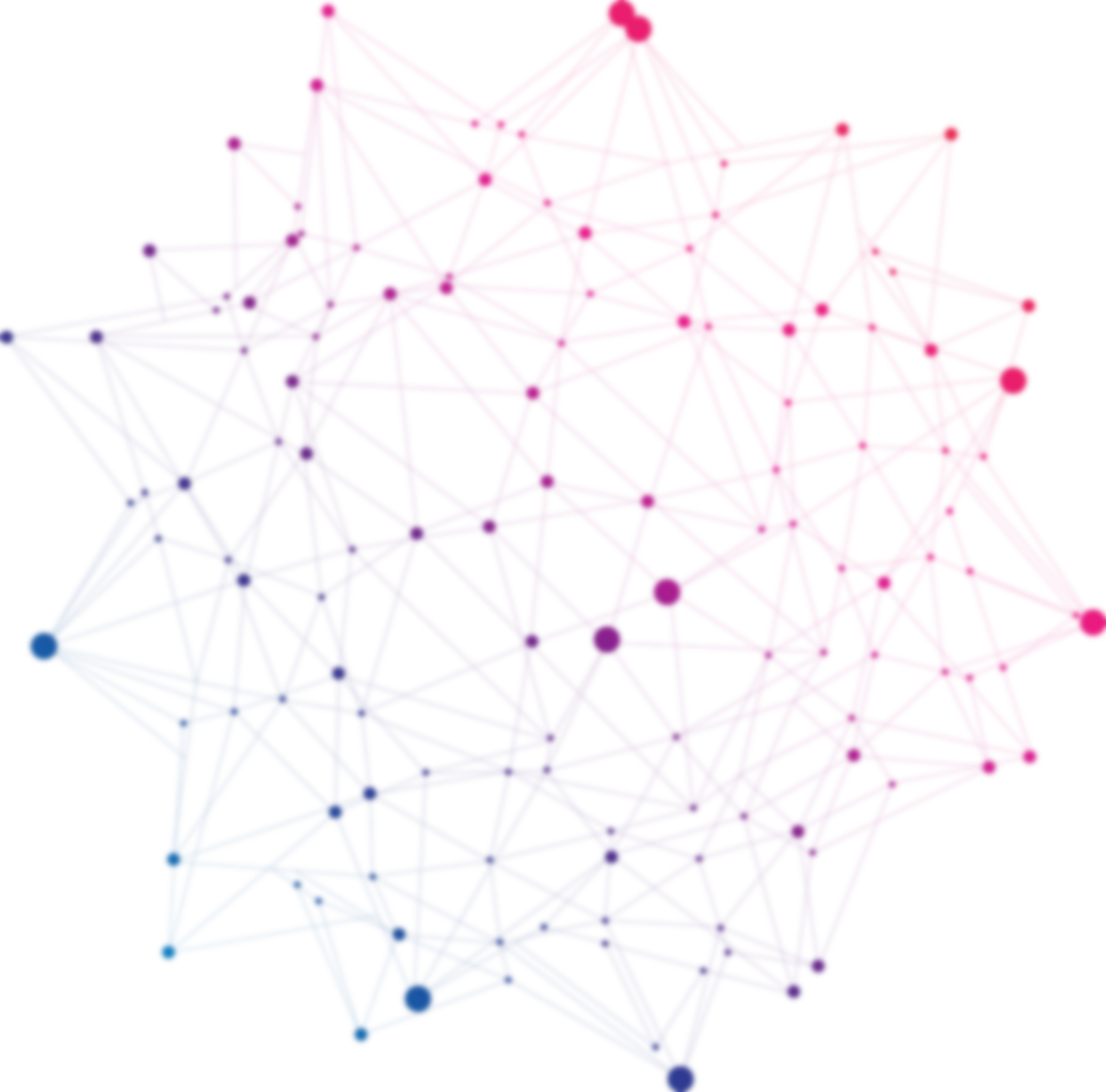
Deloitte



Adding A New Data Quality Rule in SEMOSS

Revision History

|  |  |  |
| --- | --- | --- |
| Version | Comments | Editor |
| 0.1 | Created Document. | Max Gerlach |
| 0.2 | Format Changes and some minor edits | Randall Brown |

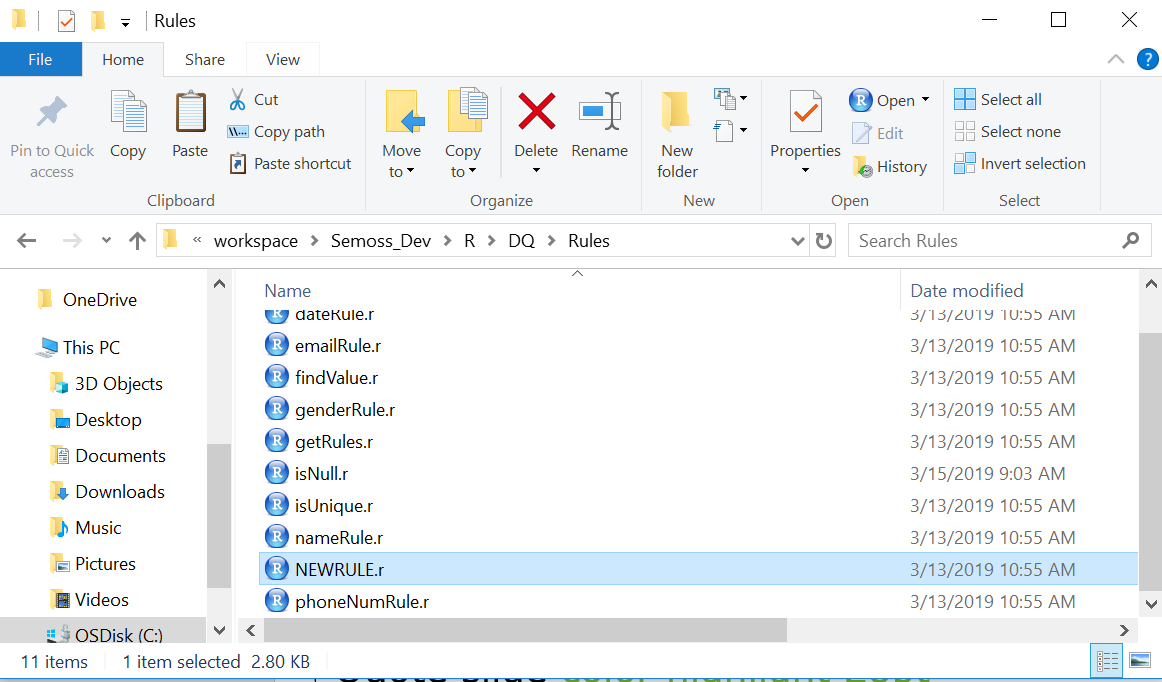
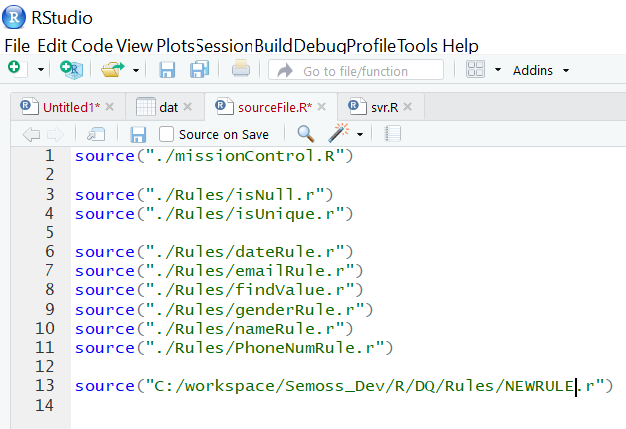
Table of Contents

[Adding New R File 1](#_Toc4146035)

[Adding the New Rule to Mission Control 1](#_Toc4146036)

[Formatting the R script to work within Data Quality Framework 2](#_Toc4146037)

# Adding New R File

1. Start by moving the newly created R file into C:\workspace\Semoss\_Dev\R\DQ\Rules directory with a name that matches the functionality of your rule. (e.g. *isNull.r*, *isUnique.r*)
2. Open *sourceFile.R* with RStudio (C:/workspace/Semoss\_Dev/R/DQ/sourceFile.R)
3. Below all the other lines of code add:

“ *source(“C:/workspace/Semoss\_Dev/R/DQ/Rules/NEWRULE.r”)* ”

1. Save sourceFile.R and close the file.

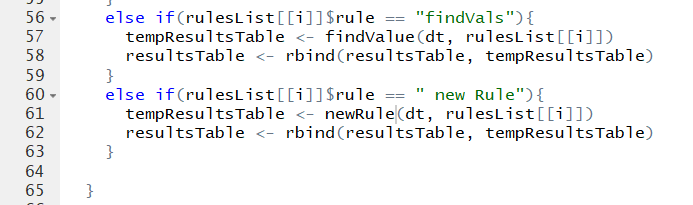
# Adding the New Rule to Mission Control

1. Open *missionControl.R* with RStudio (C:/workspace/Semoss\_Dev/R/DQ/missionControl.R)
2. After the last ‘*else if’* statement in the *rulesList* statements, add the following lines of code:

*else if(rulesList[[i]]$rule == " YOUR\_RULE\_IDENTIFIER"){*

*tempResultsTable <- YOUR\_RULE\_FUNCTION\_NAME(dt, rulesList[[i]])*

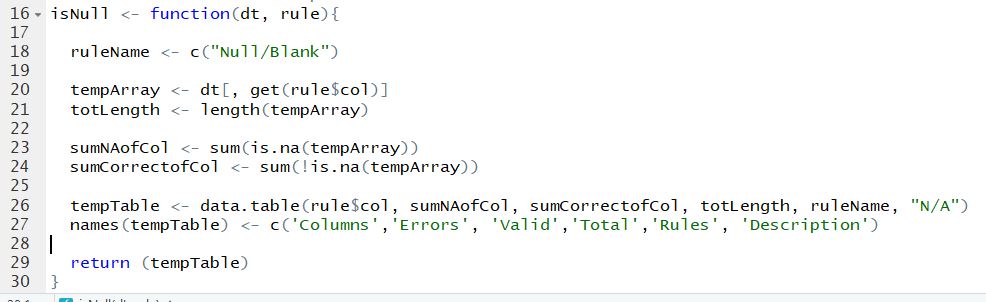
*resultsTable <- rbind(resultsTable, tempResultsTable)*

 *}*

Note: This statement calls your new function when a ‘rule identifier’ like ‘findVals’ gets passed from SEMOSS via Pixel to R.

# Formatting the R script to work within Data Quality Framework

1. If the new rule makes use of regex to compare values to a desired input format, note the *dateRule.r*, *nameRule.r*, and *emailRule.r* files.
   1. they are all almost identical besides the:
      1. *ruleName* variable, which should be set to a string of what you would like your outputted rule name to be
      2. *xxxxxErrorArray* variable name, which is a mirror of the column being checked filled with Booleans depending on whether the corresponding element are valid or invalid depending on the rule
      3. name of the function that returns the desired regex and the regexes therein
   2. Follow this format, updating it as needed for your rule.
2. Otherwise, note the parameters passed into *missionControl.R* for you to work with
   1. *rulesList* is the list of rules, which are to be executed and contain all the data for the rule
      1. a rule includes:
         1. the rule indicator, which *missionControl* uses to call the different functions
         2. the column on which to operate the rule
         3. an options variable, which can be anything else your rule needs to execute
   2. *dt* is the data table passed from SEMOSS, with the data to be analyzed
   3. *resultsTable* is the table populated with the results of the data quality rule(s)
      1. this can be passed in if you’ve already run previous rules and would like to append rule outputs or will be freshly created if nothing is passed in
3. Your R script needs to return a table with the results and proper headers so it can be bound to the main table which can be set up as such:
   1. *tempTable <- data.table(rule$col, numInvalid, numValid, totLength, ruleName, description)*
   2. *names(tempTable) <- c('Columns', 'Errors', 'Valid', 'Total', 'Rules', 'Description')*
   3. this assumes you’re storing the number of invalid elements in a variable called ‘*numInvalid*,’ the valid elements in a variable called ‘*numValid*,’ the total length of the column in a variable called ‘*totLength*,’ the name of the rule you’d like to be put in the data table in a variable called ‘*ruleName*,’ and the description in a variable called ‘description.’



1. Finally, return your table and debug if needed.