**BCQ Engine Clean-up Code Documentation**

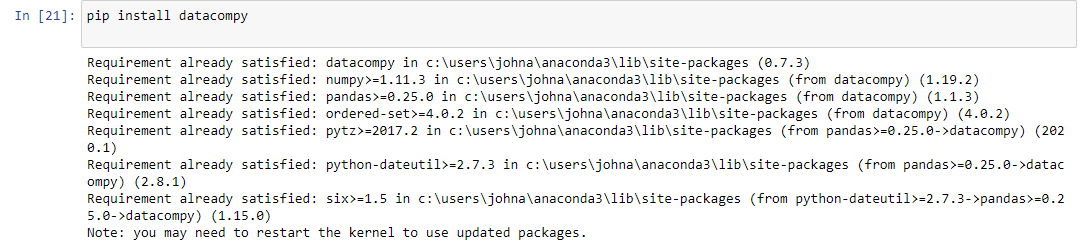
# **Code Overview**

This python code incorporates the python package datacompy which enables the comparison of two data tables through generating comprehensible reports consisting of statistics about similarities and differences between the two tables. Thus, this will help the cross-referencing process and the creation of the bcq clean-up report.

datacompy documentation : https://capitalone.github.io/datacompy/api/modules.html

# **Code**

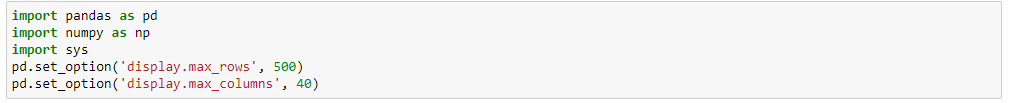
# Installing DataCompy



* Installs datacompy which is the package used to compare two dataframes.

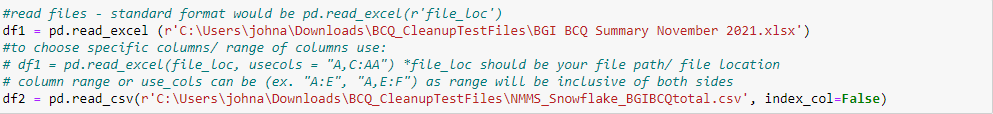
# 

# Importing Necessary Libraries and setting maximum rows and columns to be displayed



* Import pandas and numpy which are libraries primarily used for data science and analytics.
* Importing sys to be able to write outputs to files
* This part also includes setting the maximum number of displayed rows to 500 and displayed columns to 40

# Read Input Files

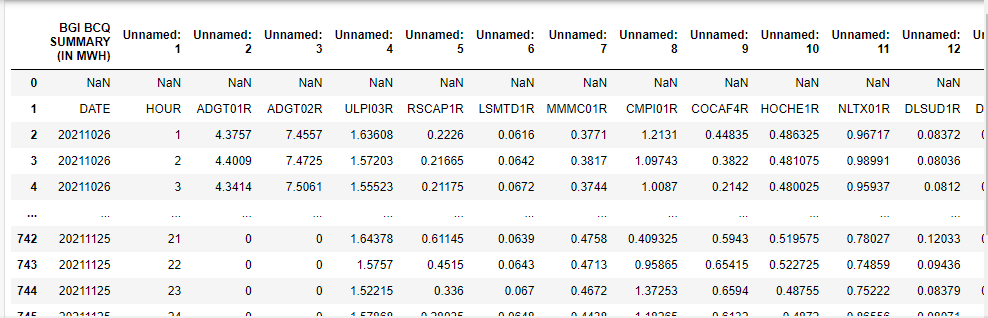


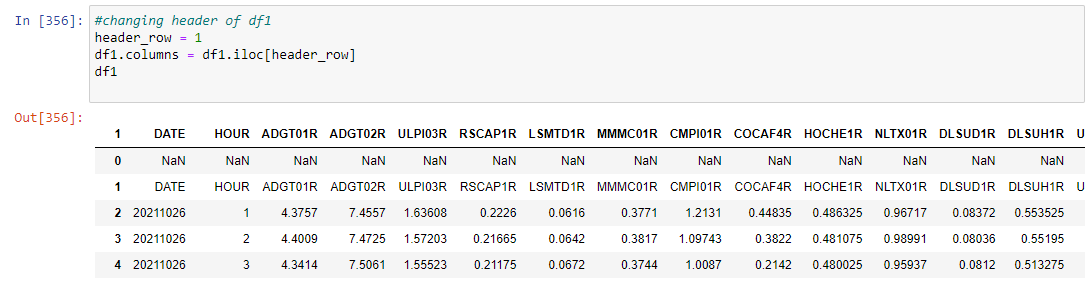
* Format inside the parentheses should always be **r‘file location’**

# Transforming Data Table 1/ Manual File Table

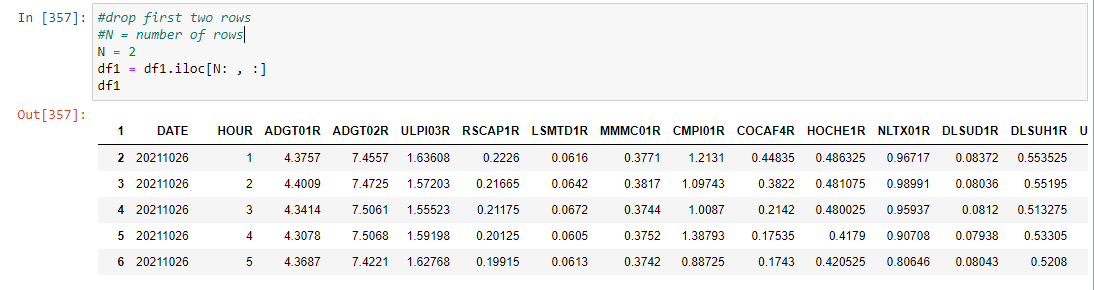
## Changing header of df1/Manual Files

* This is the first part of the data transformation and table manipulation process
* Since the current header of the manual files are as follows:



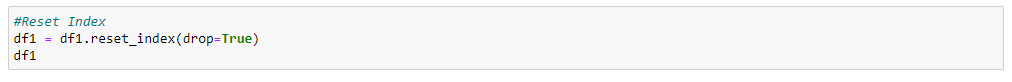
* The header location has to be changed and is done as shown below
* As seen in the image above, the header needed is from the row with an index 1 which is why header\_row = 1 is inputted.
  + Index number is seen in the leftmost column
* 

## Dropping first two rows of the table

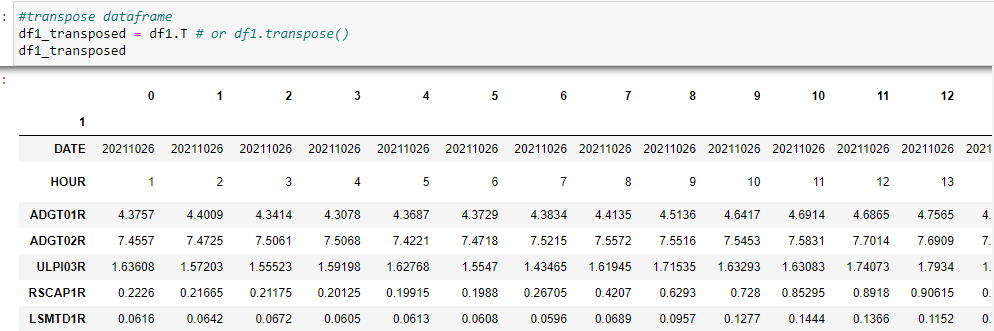


* As seen from the image in the previous step, the first two rows contain unnecessary data which is why there is a need to drop or remove these.
* This part of the code would do so
  + N would be equal to the number of rows to be dropped

## Reset Index and Transpose Current Table



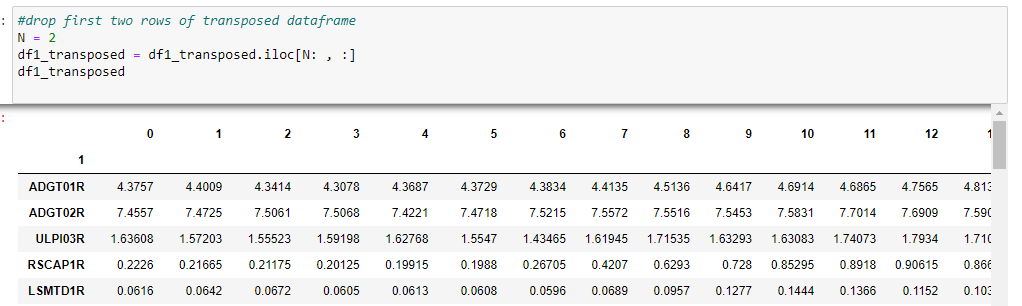
* There is a need to reset the index in order to ensure that the data table would still have consecutive indices thus making it easier to work with and compare with another table.



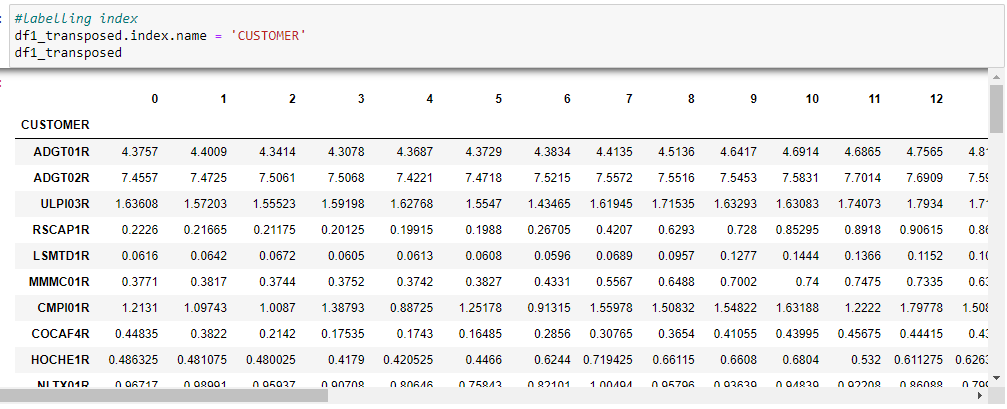
* Transpose columns to rows.

## Dropping first two rows of transposed data frame

* Again as seen from the previous image, the first two rows of the table are unnecessary which is why there is a need to remove these rows.



## Setting an Index Name



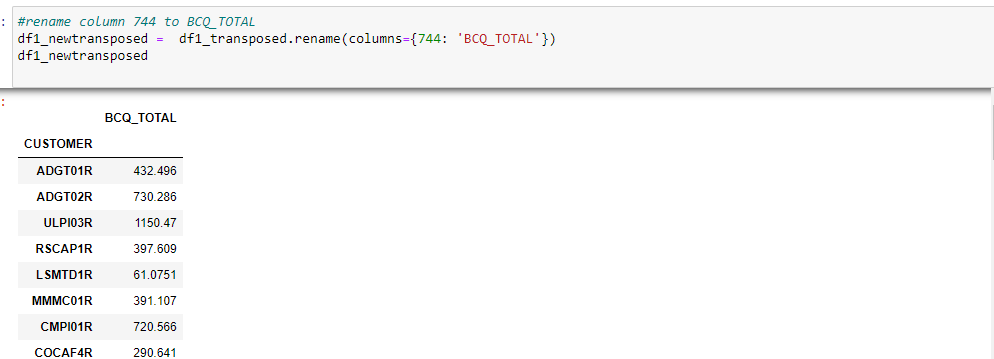
* Set the index name to ‘customer’ so that the customer ID of each customer would be the new index instead of the row number.
  + This part would help us in data table comparing later on

## Get only the last column in the table



* Since the table would only need data from the column 744 or the “Customer Total BCQ”, we can then drop all the other columns so that Customer and column 744 are the only ones presented.

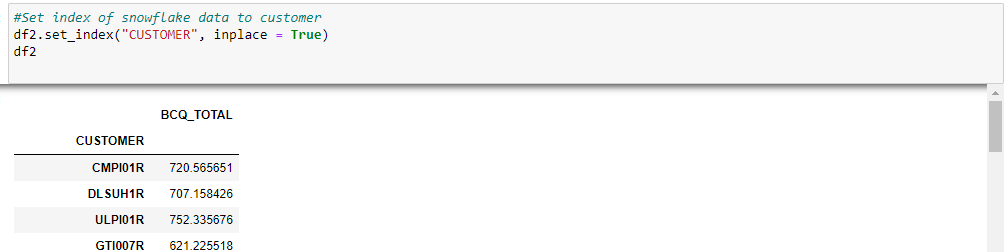
## Rename column 744 to “BCQ\_Total”



* Name column 744 to bcq\_total
* This is to ensure that the column name matches with the column name of the snowflake table as it is necessary that the two tables being compared would have the same schema.

# Transforming Data Table 2 / Snowflake Data Table

## Set Index of data table



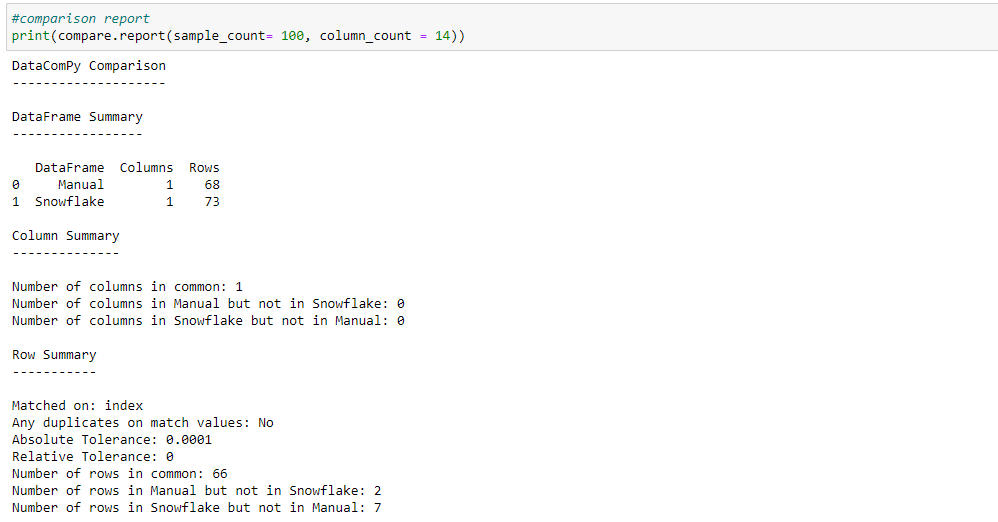
* Set the index to “Customer” so that it would have the same index as the first table and the row number columns are removed from the current table.

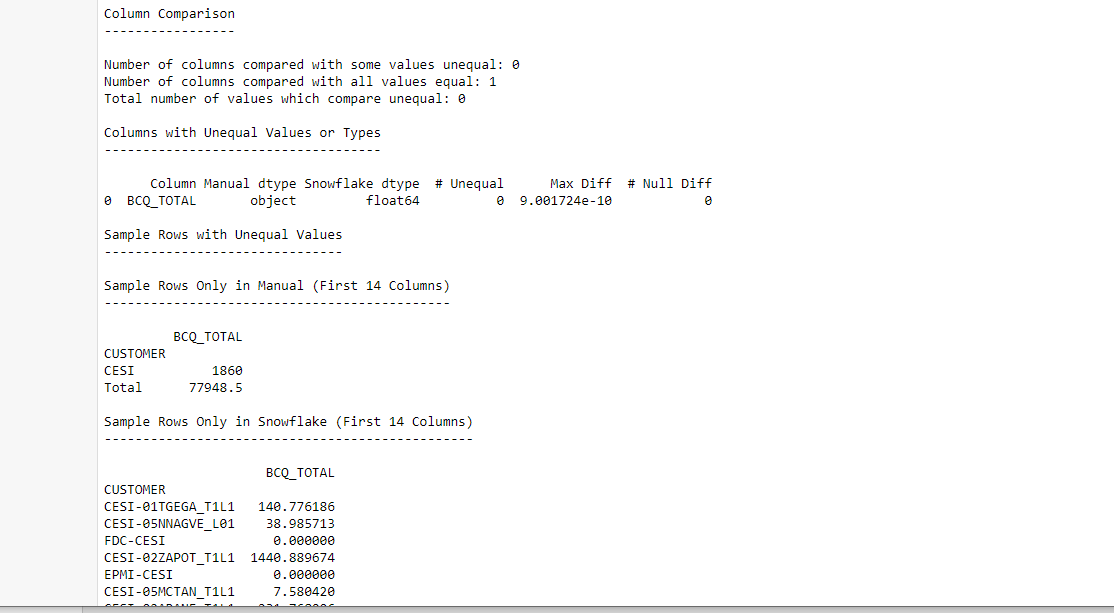
# Comparing the two Dataframes through Datacompy



* The above code is a comparison class use to compare whether the two dataframes are equal
  + Notable parameters include:
    - The names of the two dataframes. In this case df1\_newtransposed and df2
    - At what column to join the two data frames. In this case it is on the index or on\_index = True
    - String names for the two data frames.
      * df1\_name and df2\_name (Manual and Snowflake)

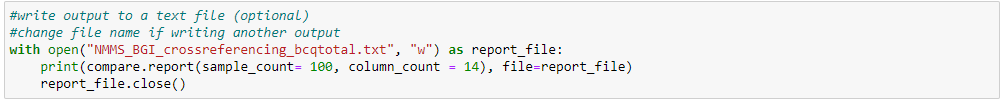
# Print Comparison Report





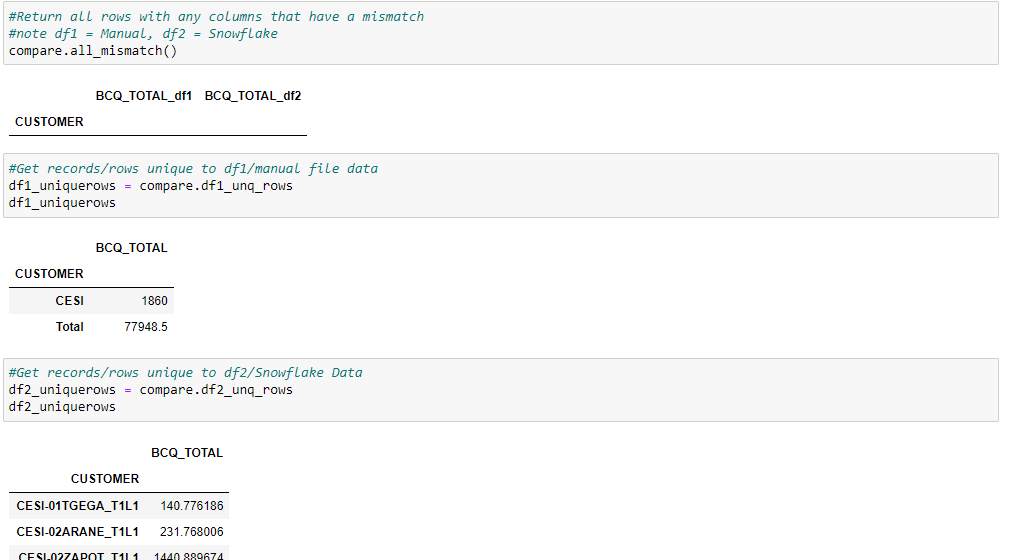
* Generate a print Output containing statistics about the similarities and differences of the two tables
* Seen above are the sample statistics.

# Write Output to Text File



* The following writes the generated or printed output in the previous step to a text or .txt file which will be saved in your local files.
* Note that it is important to remember to always change the file name if creating a new output file that makes of two new and different input files
  + For example if using manual files and snowflake data from EDC, it would be important to change “NMMS\_BGI…..\_bcqtotal.txt” to “NMMS\_EDC…..\_bcqtotal.txt”.

# Other Optional Functions for Viewing Purposes



* compare.all.mismatch()
  + Return all rows with mismatches in data
* Compare.df1\_ung\_rows
  + Return only rows unique to df1/manual file data table
* Compare.df2\_ung\_rows
  + Return only rows unique to df2/snowflake data table