**Table of Contents**

Data Quality Issues

Summary and Overview

File Inventory Status

Top 15 Users

Total and Average File Creation by Day

File Updated Distribution in Days

File Updated Lifetime

Executive Summary

> summary(data\_analyst\_assignment\_csv)

FILE\_ID USER CREATED

Length:102133 Length:102133 Min. :2019-08-01 00:00:00

Class :character Class :character 1st Qu.:2019-08-20 00:00:00

Mode :character Mode :character Median :2019-09-12 00:00:00

Mean :2019-09-13 07:57:58

3rd Qu.:2019-10-07 00:00:00

Max. :2019-10-31 00:00:00

UPDATED DELETED VERSION

Min. :2019-08-01 00:00:00 Min. :1970-01-01 00:00:00 Length:102133

1st Qu.:2019-09-09 00:00:00 1st Qu.:2019-09-05 00:00:00 Class :character

Median :2019-10-12 00:00:00 Median :2019-10-06 00:00:00 Mode :character

Mean :2019-10-12 14:40:38 Mean :2019-04-25 01:06:03

3rd Qu.:2019-11-12 00:00:00 3rd Qu.:2019-11-01 00:00:00

Max. :2020-02-04 00:00:00 Max. :2020-02-04 00:00:00

NA's :73795

**Data Quality Issues:**

The following are the two most glaring data quality issues that catch the attention immediately.

The “Version” column has “NA” data, in addition, there are some alphanumeric values that exist in that column also. In order to make that column purely numeric, these rows need to be converted to NULL. Another option is to create a new column in the data frame and convert all these values to NULL so that these values don’t impact the average calculation.

data\_issues\_VERSION\_column <- data\_analyst\_assignment\_csv[grep("[a-zA-Z]",data\_analyst\_assignment\_csv$VERSION),]

> unique(data\_issues\_VERSION\_column$VERSION)

[1] "pp" "fgT" "1pp" "4pp" "ffR" "nnJ" "3pp" "Gty" "ddR" "2pp"

Here is the quick way to create a new dataframe with “VERSION” column only including the numeric data, and then converting it to an integer column. It is a three step process. The first is to copy the alphanumeric character from the “VERSION” into a new dataframe, then comparing the newly created dataframe with the existing dataframe and excluding the newly created dataframe’s fileIDs from the existing dataframe. Lastly, copy that into a new data frame. Now, the newly dataframe will have all the VERSION data as number (it only needs to converted to integer explicitly via the as.integer function call)

files\_with\_VERSION\_alpha\_characters <- data\_analyst\_assignment\_csv[grep("[a-zA-Z]",data\_analyst\_assignment\_csv$VERSION),]

VERSION\_column\_number <- data\_analyst\_assignment\_csv[!((data\_analyst\_assignment\_csv$FILE\_ID) %in% (files\_with\_VERSION\_alpha\_characters$FILE\_ID)),]

VERSION\_column\_number$VERSION <- as.integer(VERSION\_column\_number$VERSION)

Secondly, the “deleted” column has some values which are earlier than the “created at” which don’t make sense. There are 259 of those rows. They need to be eliminated when performing the average file duration calculation.

> subset(data\_analyst\_assignment\_csv,DELETED < CREATED,select=(1:6))

# A tibble: 259 x 6

FILE\_ID USER CREATED UPDATED DELETED VERSION

*<chr>* *<chr>* *<dttm>* *<dttm>* *<dttm>* *<chr>*

1 rrT0625 2853751 2019-08-01 00:00:00 2019-11-19 00:00:00 1970-01-01 00:00:00 26

2 KLp3685 2728736 2019-08-01 00:00:00 2019-08-01 00:00:00 1970-01-01 00:00:00 2

3 KLp4075 3161776 2019-08-01 00:00:00 2019-12-09 00:00:00 1970-01-01 00:00:00 2

4 KLp4290 262451 2019-08-01 00:00:00 2019-08-21 00:00:00 1970-01-01 00:00:00 2

5 ffR12445 43539 2019-08-02 00:00:00 2019-10-14 00:00:00 1970-01-01 00:00:00 3

6 ffR32025 872068 2019-08-05 00:00:00 2019-09-19 00:00:00 1970-01-01 00:00:00 13

7 ffR36495 2629501 2019-08-05 00:00:00 2019-12-12 00:00:00 1970-01-01 00:00:00 17

8 ffR36505 2871311 2019-08-05 00:00:00 2019-12-04 00:00:00 1970-01-01 00:00:00 5

9 ffR36680 2871311 2019-08-05 00:00:00 2019-12-04 00:00:00 1970-01-01 00:00:00 3

10 ffR48990 180880 2019-08-06 00:00:00 2019-08-06 00:00:00 1970-01-01 00:00:00 2

# ... with 249 more rows

> subset(data\_analyst\_assignment\_csv,DELETED < CREATED,select=(1:6)) %>% length()

[1] 6

> subset(data\_analyst\_assignment\_csv,DELETED < CREATED,select=(1:6)) %>% dim()

[1] 259 6

**Lastly, these four files belong to “Unknown” user**

> data\_analyst\_assignment\_csv[is.na(data\_analyst\_assignment\_csv$USER),]

# A tibble: 4 x 7

FILE\_ID USER CREATED UPDATED DELETED VERSION

*<chr>* *<chr>* *<dttm>* *<dttm>* *<dttm>* *<chr>*

1 180360~ NA 2019-09-06 00:00:00 2019-09-06 00:00:00 NA 1

2 ddR515~ NA 2019-10-22 00:00:00 2019-10-22 00:00:00 NA 1

3 ddR520~ NA 2019-10-22 00:00:00 2019-10-22 00:00:00 NA 1

4 Gty215~ NA 2019-10-31 00:00:00 2019-10-31 00:00:00 NA 1

# ... with 1 more variable: file\_created\_date *<chr>*

**Quick Overview of PostClick Data**

Here is the quick overview of PostClick Data. Barring data anomalies and data qualities issues, these numbers depict at a high level how well the data looks like.

Libraries used to analyze this data are:

* Sqldf
* Dplyr
* Lubridate
* Ggplot2

labels data\_labels

**"Total Unique Users" 10036**

**"Total Files Created" 102133**

**"Total Created Days" 92**

**"Total Files Updated" 68259**

**"Total Files Deleted" 28338**

**"Average File Life Span (in days)" 26.43829**

**"Users with Multiple files" "7213 (71%)"**

**"Users with Single file" "2822 (28%)"**

avg\_file\_lifespan <- VERSION\_column\_number %>%

filter(DELETED >= CREATED) %>%

mutate(file\_lifespan\_in\_days = difftime(as.Date(as.POSIXct(DELETED,origin="1970-01-01")),as.Date(as.POSIXct(CREATED,origin="1970-01-01")),units='days')) %>%

select(FILE\_ID,

USER,

CREATED,

UPDATED,

DELETED,

VERSION,

file\_lifespan\_in\_days) %>%

summarize(average\_file\_lifespan = mean(file\_lifespan\_in\_days,na.rm=TRUE))

labels <- c("Total Unique Users",

"Total Files Created",

"Total Created Days",

"Total Files Updated",

"Total Files Deleted",

"Average File Life Span (in days)",

"Users with Multiple files",

"Users with Single file")

data\_labels <- c(length(unique(data\_analyst\_assignment\_csv$USER)),

length(unique(data\_analyst\_assignment\_csv$FILE\_ID)),

length(unique(data\_analyst\_assignment\_csv$CREATED)),

dim(subset(data\_analyst\_assignment\_csv,UPDATED > CREATED))[1],

dim(data\_analyst\_assignment\_csv[!is.na(data\_analyst\_assignment\_csv$DELETED),])[1],

avg\_file\_lifespan[1],

paste(multi\_single\_file\_user[,1]," (",multi\_single\_file\_user[,3],"%)",sep=""),

paste(multi\_single\_file\_user[,2]," (",multi\_single\_file\_user[,4],"%)",sep="")

)

cbind(labels,data\_labels)

multi\_single\_file\_user <- sqldf("SELECT

COUNT(CASE WHEN total\_files > 1 THEN user\_id ELSE NULL END) AS total\_users\_with\_multiple\_files,

COUNT(CASE WHEN total\_files = 1 THEN user\_id ELSE NULL END) AS total\_users\_with\_one\_file,

100\*COUNT(CASE WHEN total\_files > 1 THEN user\_id ELSE NULL END) / COUNT(DISTINCT user\_id) AS multiple\_files\_user\_per,

100\*COUNT(CASE WHEN total\_files = 1 THEN user\_id ELSE NULL END) / COUNT(DISTINCT user\_id) AS single\_file\_user\_per

FROM

(

SELECT

USER AS user\_id,

COUNT(DISTINCT FILE\_ID) AS total\_files

FROM data\_analyst\_assignment\_csv

GROUP BY 1

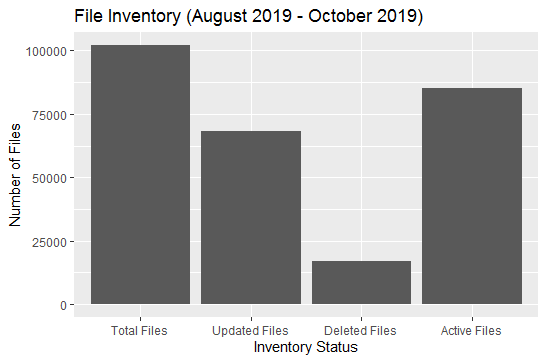
) main")

**File Inventory Status**:

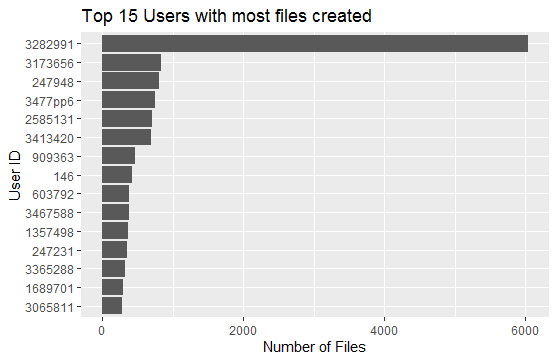
It is always insightful to look at the current inventory at hand. Here is the high level file inventory status from August 2019 to October 2019.

The assumption for “Updated” files is only when the UPDATED date is greater than the CREATED date. If both dates are equal, then the files are assumed not be “updated”.

Active count excludes the files which were “Deleted”.



|  |
| --- |
| **Top Users**:  In order to see how users are utilizing the files, this was the most basic and the easiest insight that could be looked at.  The chart quite clearly showed an outlier where only one user’s frequency is relatively large compared to the others.  > top\_15\_users <- sqldf("SELECT  + USER AS user\_id,  + COUNT(DISTINCT FILE\_ID) AS total\_files  + FROM data\_analyst\_assignment\_csv  + GROUP BY user\_id  + ORDER BY 2 DESC  + LIMIT 15  + ")  >  >  > plot\_top\_15\_users <- ggplot(top\_15\_users, aes(reorder(x=user\_id,total\_files), y=total\_files)) +  + geom\_bar(stat = "identity") +  + coord\_flip() +  + ggtitle("Top 15 Users with most files created") +  + ylab("Number of Files") +  + xlab("User ID")  > print(plot\_top\_15\_users) |
|  |
| |  | | --- | |  | |



**Total and Average File Creation Per Day**:

This trend paints an interesting but steady picture. Barring a sudden peak in the mid of August, the average file creation per day looks pretty steady and consistent.

> plot\_dates\_file\_created\_at <- ggplot(dates\_file\_created\_at, aes(x=date(date\_file\_created), y=total\_files\_created)) +

+ geom\_line() +

+ ggtitle("Total Files by Day") +

+ ylab("Number of Files Created") +

+ xlab("Day") +

+ scale\_x\_date(date\_labels = "%Y (%b)")

> print(plot\_dates\_file\_created\_at)

> library(lubridate)

> dates\_file\_created\_at <- data\_analyst\_assignment\_csv %>%

+ mutate(date\_file\_created = as.POSIXct(CREATED,origin="1970-01-01")) %>%

+ select(date\_file\_created,FILE\_ID) %>%

+ group\_by(date\_file\_created) %>%

+ summarize(total\_files\_created = n\_distinct(FILE\_ID)) %>%

+ arrange(desc(total\_files\_created))

>

> plot\_dates\_file\_created\_at <- ggplot(dates\_file\_created\_at, aes(x=date(date\_file\_created), y=total\_files\_created)) +

+ geom\_line() +

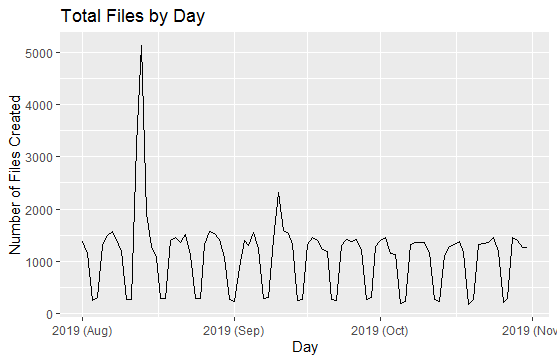
+ ggtitle("Total Files by Day") +

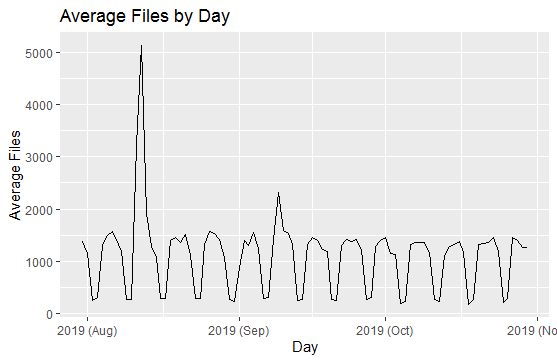
+ ylab("Number of Files Created") +

+ xlab("Day") +

+ scale\_x\_date(date\_labels = "%Y (%b)")

> print(plot\_dates\_file\_created\_at)





**Total File Creation Per Week Day**:

This gives another interesting insight as to what week days are the busiest for file creation.

> print(plot\_week\_day\_file\_created\_at)

> library(lubridate)

> week\_day\_file\_created\_at <- data\_analyst\_assignment\_csv %>%

+ mutate(date\_file\_created = as.Date(as.POSIXct(CREATED,origin="1970-01-01"))) %>%

+ select(date\_file\_created,FILE\_ID) %>%

+ group\_by(date\_file\_created) %>%

+ summarize(total\_files\_created = n\_distinct(FILE\_ID)) %>%

+ arrange(desc(total\_files\_created))

>

> plot\_week\_day\_file\_created\_at <- ggplot(week\_day\_file\_created\_at, aes(x=wday(date\_file\_created,label=TRUE), y=total\_files\_created,1,group=1)) +

+ stat\_summary(fun.y = sum, geom = "bar") +

+ ggtitle("Total Files by Week Day") +

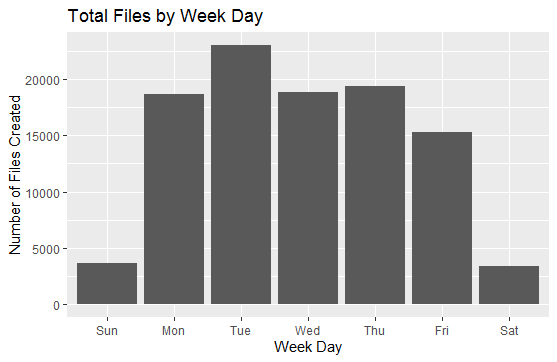
+ ylab("Number of Files Created") +

+ xlab("Week Day")

>

>

> print(plot\_week\_day\_file\_created\_at)



**File Updated Distribution in Days**:

This analysis gives another insight as to how long it takes for a file to get updated.

Majority of the file updates happen on “Day 0”.

> file\_update\_time\_distribution <- data\_analyst\_assignment\_csv %>%

+ mutate(time\_in\_days = difftime(as.Date(as.POSIXct(UPDATED,origin="1970-01-01")),as.Date(as.POSIXct(CREATED,origin="1970-01-01")),units='days')) %>%

+ select(time\_in\_days,FILE\_ID) %>%

+ group\_by(time\_in\_days) %>%

+ summarize(total\_files\_updated = n\_distinct(FILE\_ID)) %>%

+ filter(total\_files\_updated > 0)

> plot\_file\_update\_time\_distribution <- ggplot(file\_update\_time\_distribution, aes(x=time\_in\_days, y=total\_files\_updated)) +

+ geom\_line() +

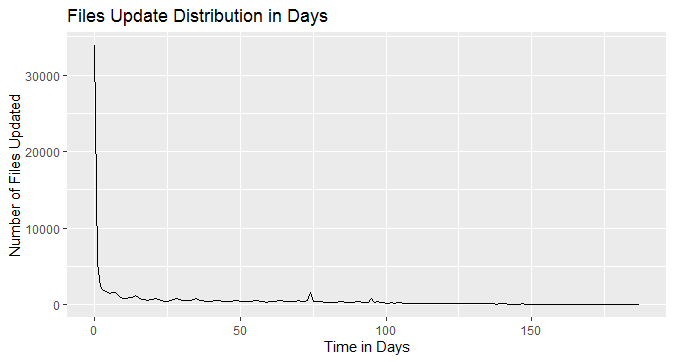
+ ggtitle("Files Update Distribution in Days") +

+ ylab("Number of Files Updated") +

+ xlab("Time in Days")

>

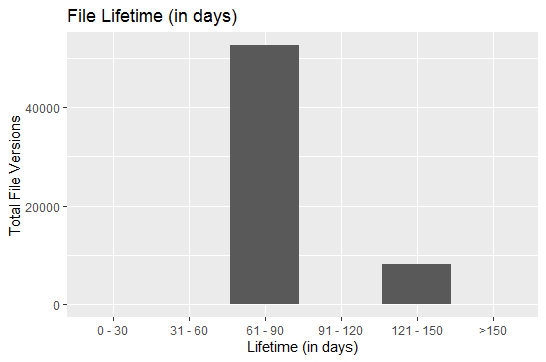
> print(plot\_file\_update\_time\_distribution)



**File Update Lifetime:**

After converting the “VERSION” into pure integer column, the trend (histogram) is to see the number of file versions and compare the lifetime in days.

Clearly, after bucketing the number of days, it is apparent that the lifespan is fall under 61-90 and 121 – 150. This differs from the “average file lifetime span” since it doesn’t average out the number of days.



> total\_files\_update\_VERSION\_by\_days <- VERSION\_column\_number %>%

+ # filter(FILE\_ID %in% ('17849000')) %>%

+ mutate(file\_lifespan\_in\_days = difftime(as.Date(as.POSIXct(DELETED,origin="1970-01-01")),as.Date(as.POSIXct(CREATED,origin="1970-01-01")),units='days')) %>%

+ mutate(lifespan\_label = case\_when(file\_lifespan\_in\_days >=0 & file\_lifespan\_in\_days <=30 ~ "0 - 30",

+ file\_lifespan\_in\_days >=31 & file\_lifespan\_in\_days <=60 ~ "31 - 60",

+ file\_lifespan\_in\_days >=61 & file\_lifespan\_in\_days <=90 ~ "61 - 90",

+ file\_lifespan\_in\_days >=90 & file\_lifespan\_in\_days <=120 ~ "91 - 120",

+ file\_lifespan\_in\_days >=121 & file\_lifespan\_in\_days <=150 ~ "121 - 150",

+ TRUE ~ ">150"

+ )

+ ) %>%

+ select(FILE\_ID,

+ USER,

+ CREATED,

+ UPDATED,

+ DELETED,

+ VERSION,

+ # file\_lifespan\_in\_days,

+ lifespan\_label) %>%

+ group\_by(lifespan\_label) %>%

+ summarize(total\_files\_updated = sum(VERSION))

`summarise()` ungrouping output (override with `.groups` argument)

>

> total\_files\_update\_VERSION\_by\_days$lifespan\_label <- factor(total\_files\_update\_VERSION\_by\_days$lifespan\_label,levels= c("0 - 30","31 - 60","61 - 90","91 - 120","121 - 150", ">150"),ordered=TRUE)

> total\_files\_update\_VERSION\_by\_days$total\_files\_updated <- ifelse(is.na(total\_files\_update\_VERSION\_by\_days$total\_files\_updated),0,total\_files\_update\_VERSION\_by\_days$total\_files\_updated)

> total\_files\_update\_VERSION\_by\_days[order(total\_files\_update\_VERSION\_by\_days$lifespan\_label),]

# A tibble: 6 x 2

lifespan\_label total\_files\_updated

*<ord>* *<dbl>*

1 0 - 30 0

2 31 - 60 0

3 61 - 90 52637

4 91 - 120 0

5 121 - 150 8058

6 >150 0

>

>

>

> plot\_total\_files\_update\_VERSION\_by\_days <- ggplot(total\_files\_update\_VERSION\_by\_days, aes((x=total\_files\_updated), y=lifespan\_label)) +

+ geom\_bar(stat = "identity") +

+ coord\_flip() +

+ ggtitle("File Lifetime (in days)") +

+ ylab("Lifetime (in days)") +

+ xlab("Total File Versions")

> print(plot\_total\_files\_update\_VERSION\_by\_days)

**Executive Summary**

* The data is generally clean and easy to comprehend and analyze. The only issue was the field “delimiter”. I used the global replace function to convert the delimiter into a single comma.
* As for the data quality issues, the “Deleted” column has some dates where were earlier than the file creation date. For “average file span” calculation, I needed to exclude those rows.
* File inventory status is another key metric to see the funnel as a whole. It gives us a nice high level picture of what exists in the data file.
* Looking at the most frequent user data, we have got an interesting finding. One particular user generated a lot of files. It can easily be classified as an outlier since most other users with more than one file seem very consistent in this dataset.
* Users with multiple files have are clearly on a higher side (71%).
* Average file per day shows a sudden spike in the middle of August.