SOT/OTS Weekly Process

```
library(dplyr)
library(readr)
library(RODBC)
library(formattable)
library(RJDBC)
library(rChoiceDialogs)
```

After loading the required libraries, create a connection to EDWP using the *RODBC* library. Once you have stored your username and password as ***my_uid*** and ***my_pwd*** ; and created a DSN, connect with a string similar to this (You may need to change the below if you gave your DSN an different name). Then verify that we have successfully connected by performing a query on the dbcinfo table.

Set up

```
# Create RODBC connection ----
my_connect <- odbcConnect(dsn= "IP EDWP", uid= my_uid, pwd= my_pwd)
# sqlTables(my_connect, catalog = "EDWP", tableName  = "tables")
sqlQuery(my_connect, query = "SELECT  * from dbc.dbcinfo;")
```

```
##                 InfoKey    InfoData
## 1               VERSION 14.10.07.10
## 2               RELEASE 14.10.07.09
## 3 LANGUAGE SUPPORT MODE    Standard
```

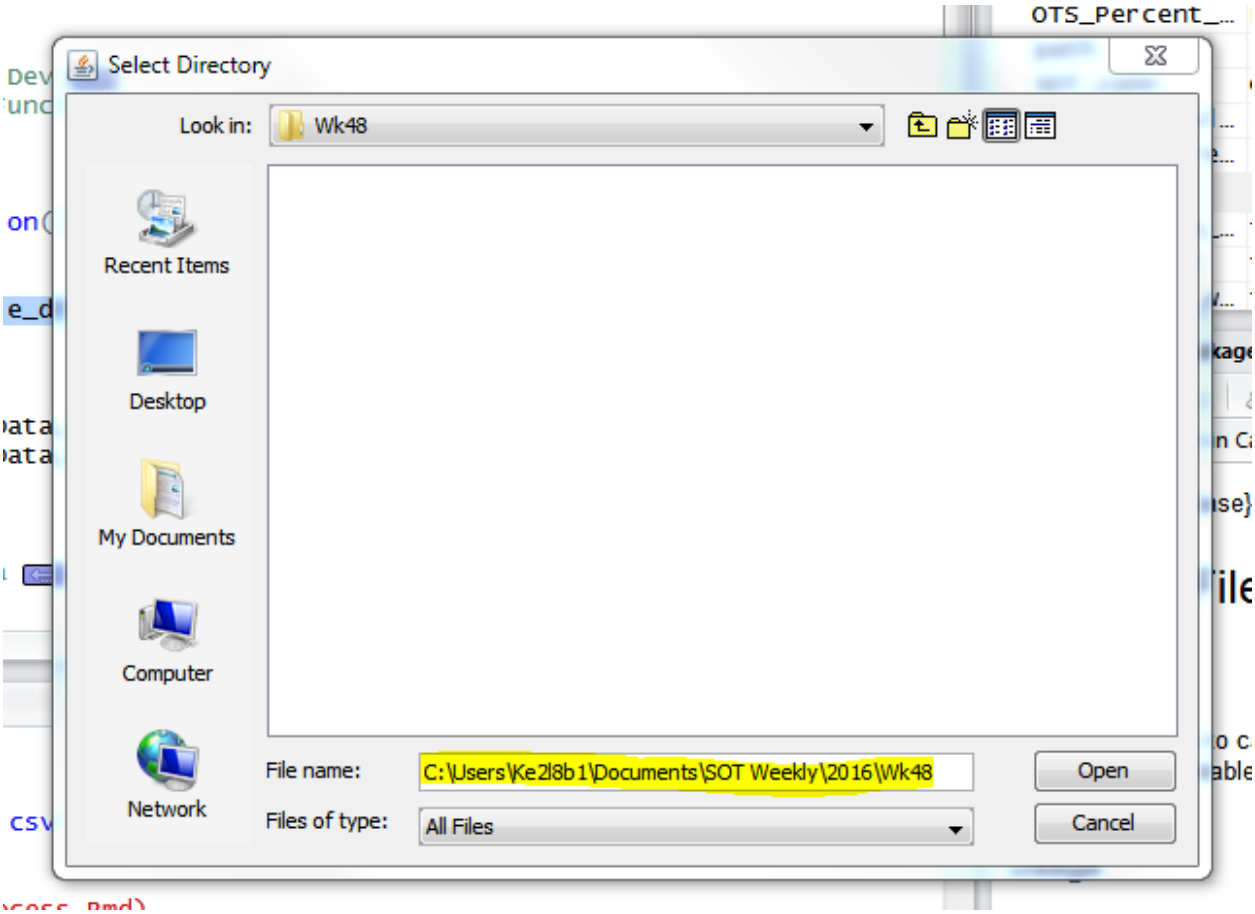
Now we need to create a few functions and store a few constants for later use.

```
# Setup Environment Variables/Functions----
prompt_for_week <- function()
{
  n <- readline(prompt="Enter Week number: ")
  return(as.integer(n))
}

choose_file_directory <- function()
{
  v <- jchoose.dir()
  return(v)
}
```

We want to store our directory path as a string. The easiest way to do this is by opening a Java based file chooser (sometimes this has to be ran twice. It intermittently fails the first time). The code below will store a path to a directory of your choice.

```
SOT_OTS_directory <- choose_file_directory()
```



Now enter the week for which you are reporting. This will prompt you at the console.

```
EOW <- prompt_for_week()
```

Pulling data

All tables are built from two *Master* tables. We need to create them now via query to EDW.

```
# Create Master Objects ----
SOT_Master <- sqlQuery(my_connect,
                       query = "SELECT  * from SRAA_SAND.VIEW_SOT_MASTER;")

OTS_Master <- sqlQuery(my_connect,
                           query = "SELECT  * from SRAA_SAND.VIEW_OTS_MASTER;")
```

Close the connection:

```
# Close connection ----
close(my_connect)
```

The tables, we just created, have a field labeled *Data_Pulled* that is populated during *CREATE TABLE* in Teradata. It indicates when the data was pulled from the IUF tables. Let's store the first value from each table for reference.

```
SOT_Data_Pulled <- SOT_Master$Data_Pulled[1]
OTS_Data_Pulled <- OTS_Master$Data_Pulled[1]
```

Generate Metadata

Let's now create and store some Metadata about our raw data frames...

```
# Create/write Summary Metadata ----
SOT_Master_Summary <- as.data.frame(summary(SOT_Master))
OTS_Master_Summary <- as.data.frame(summary(OTS_Master))


write_csv(SOT_Master_Summary, path = paste(
  SOT_OTS_directory,
  paste('SOT_Master_RAW_Metadata_WK', EOW, '.csv', sep = ""),
  sep = '/'
  ))
  write_csv(OTS_Master_Summary, path = paste(
  SOT_OTS_directory,
  paste('OTS_Master_RAW_Metadata_WK', EOW, '.csv', sep = ""),
  sep = '/'
  ))
```

...and also write our masters to .csv.

```
# Write Raw files to .csv ----
write_csv(SOT_Master, path = paste(SOT_OTS_directory,  'SOT_Master_Raw.csv', sep = '/' ))
write_csv(OTS_Master, path = paste(SOT_OTS_directory,  'OTS_Master_Raw.csv', sep = '/' ))
```

Now let's save the Raw objects. These will be saved as binary files that can be quickly loaded into R. This will allow us to reproduce our results, at any time, using the same raw data.

```
# Save Raw objects ----
save(SOT_Master, file = paste(SOT_OTS_directory,  'SOT_Master_object.rtf', sep = .Platform$file.sep))
save(OTS_Master, file = paste(SOT_OTS_directory,  'OTS_Master_object.rtf', sep = .Platform$file.sep ))
```

We usually don't include all vendors (i.e. non-apparel); nor do we include the virtual DC (JPF). Clean them up with the below.

```
# Scrub Noise from Master Objects ----
OTS_Master <- OTS_Master %>%
  filter(OTS_Master$Week <= EOW,
         !is.na(OTS_Master$DC_NAME),
         !grepl("Liberty Distribution Company", Parent_Vendor, ignore.case = TRUE),
         !grepl("dummy", Parent_Vendor, ignore.case = TRUE),
         !grepl("JPF", DC_NAME, ignore.case = TRUE))

SOT_Master <- SOT_Master %>%
  filter(SOT_Master$ShipCancelWeek <= EOW,
          !grepl("Liberty Distribution Company", Parent_Vendor, ignore.case = TRUE),
          !grepl("dummy", Parent_Vendor, ignore.case = TRUE),
          MetricShipDate <= SOT_Data_Pulled)
```

Let's also create Metadata for the current week and write it to csv for EDA purposes.

```
# Create/write Metadata for Week subset ----
SOT_Master_Summary_curr_week <-
  SOT_Master %>% filter(ShipCancelWeek == EOW) %>% summary() %>% as.data.frame()

OTS_Master_Summary_curr_week <-
  OTS_Master %>% filter(Week == EOW) %>% summary() %>% as.data.frame()
```

```
write_csv(
  as.data.frame(SOT_Master_Summary_curr_week),
  path = paste(
    SOT_OTS_directory,
    paste('SOT_Master_Metadata_curr_week', EOW, '.csv', sep = ""),
    sep = '/'))

write_csv(
  as.data.frame(OTS_Master_Summary_curr_week),
  path = paste(
    SOT_OTS_directory,
    paste('OTS_Master_Metadata_curr_week', EOW, '.csv', sep = ""),
    sep = '/'))
```

Building the Output Tables

We need to build 4 tables for output to the presentation layer. We need both a Category and Vendor view (for OTS and SOT), and need to create some new summary columns.

```
# Create Output Tables ----

# 1) OTS by Category Summary
OTS_by_Category <- OTS_Master %>%
  filter(!grepl("FRANCHISE", ReportingBrand, ignore.case = TRUE, fixed=FALSE)) %>%
  group_by(ReportingBrand, Category, Month_Number,Week, DC_NAME) %>%
  summarise(TotalUnits= sum(Units),
            OnTimeUnits = sum(Units[Lateness == "OnTime"]),
            LateUnits = sum(Units[Lateness == "Late"]),
            WtDaysLate = sum(Units[Lateness == "Late"] * Days_Late[Lateness ==
            "Late"]),
            DaysLate5 = sum(Units[Days_Late > 5 &
            Lateness == "Late"]),
            UnitsArriveLessThanNeg5 = sum(Units[(Lateness == "OnTime" |
            Lateness == "Late") & (Days_Late <= -5)]),
            UnitsArriveLessThanNeg3 = sum(Units[(Lateness == "OnTime" |
            Lateness == "Late") & (Days_Late <= -3)]),
            UnitsArriveLessThan0 = sum(Units[(Lateness == "OnTime" |
            Lateness == "Late") & (Days_Late <= 0)]),
            UnitsArriveLessThan3 = sum(Units[(Lateness == "OnTime" |
            Lateness == "Late") & (Days_Late <= 3)]),
            UnitsArriveLessThan5 = sum(Units[(Lateness == "OnTime" |
            Lateness == "Late") & (Days_Late <= 5)])) %>%
  droplevels()

# 2) OTS by Vendor Summary
OTS_by_Vendor <- OTS_Master %>%
  filter(!grepl("FRANCHISE", ReportingBrand, ignore.case = TRUE, fixed = FALSE)) %>%
  group_by(Vendor_Rank, Parent_Vendor, Month_Number, Week) %>%
  summarise(
  TotalUnits = sum(Units),
  OnTimeUnits = sum(Units[Lateness == "OnTime"]),
  LateUnits = sum(Units[Lateness == "Late"]),
  WtDaysLate = sum(Units[Lateness == "Late"] * Days_Late[Lateness == "Late"]),
```

```
  DaysLate5 = sum(Units[Days_Late > 5 &
  Lateness == "Late"]),
  UnitsArriveLessThanNeg5 = sum(Units[(Lateness == "OnTime" |
  Lateness == "Late") & (Days_Late <= -5)]),
  UnitsArriveLessThanNeg3 = sum(Units[(Lateness == "OnTime" |
  Lateness == "Late") & (Days_Late <= -3)]),
  UnitsArriveLessThan0 = sum(Units[(Lateness == "OnTime" |
  Lateness == "Late") & (Days_Late <= 0)]),
  UnitsArriveLessThan3 = sum(Units[(Lateness == "OnTime" |
  Lateness == "Late") & (Days_Late <= 3)]),
  UnitsArriveLessThan5 = sum(Units[(Lateness == "OnTime" |
  Lateness == "Late") & (Days_Late <= 5)])
  ) %>%
  droplevels()


# 3) SOT by Category Summary
SOT_by_Category <- SOT_Master %>%
  filter(!grepl("FRANCHISE", ReportingBrand, ignore.case = TRUE, fixed = FALSE)) %>%
  group_by(ReportingBrand, Category, ShipCancelMonth, ShipCancelWeek) %>%
  summarise(
  TotalUnits = sum(Units),
  OnTimeUnits = sum(Units[Lateness == "OnTime"]),
  LateUnits = sum(Units[Lateness == "Late"]),
  WtDaysLate = sum(Units[Lateness == "Late"] * DAYS_LATE[Lateness ==
  "Late"]),
  DaysLate5 = sum(Units[DAYS_LATE > 5], na.rm = TRUE),
  UnitsArriveLessThanNeg5 = sum(Units[Lateness == "OnTime" &
  DAYS_LATE <= -5]),
  UnitsArriveLessThanNeg3 = sum(Units[Lateness == "OnTime" &
  DAYS_LATE <= -2]),
  UnitsArriveLessThan0 = sum(Units[Lateness == "OnTime" &
  DAYS_LATE <= 0]),
  UnitsArriveLessThan3 = sum(Units[(Lateness == "OnTime" |
  Lateness == "Late") & DAYS_LATE <= 2]),
  UnitsArriveLessThan5 = sum(Units[(Lateness == "OnTime" |
  Lateness == "Late") & DAYS_LATE <= 5])
  ) %>%
  droplevels()

# 4) SOT by Vendor Summary
SOT_by_Vendor <- SOT_Master %>%
  filter(!grepl("FRANCHISE", ReportingBrand, ignore.case = TRUE, fixed = FALSE)) %>%
  group_by(Vendor_Rank, Parent_Vendor, ShipCancelMonth, ShipCancelWeek) %>%
  summarise(
  TotalUnits = sum(Units),
  OnTimeUnits = sum(Units[Lateness == "OnTime"]),
  LateUnits = sum(Units[Lateness == "Late"]),
  WtDaysLate = sum(Units[Lateness == "Late"] * DAYS_LATE[Lateness ==
  "Late"]),
  DaysLate5 = sum(Units[DAYS_LATE > 5], na.rm = TRUE),
  UnitsArriveLessThanNeg5 = sum(Units[Lateness == "OnTime" &
  DAYS_LATE <= -5]),
```

```
  UnitsArriveLessThanNeg3 = sum(Units[Lateness == "OnTime" &
  DAYS_LATE <= -2]),
  UnitsArriveLessThan0 = sum(Units[Lateness == "OnTime" &
  DAYS_LATE <= 0]),
  UnitsArriveLessThan3 = sum(Units[(Lateness == "OnTime" |
  Lateness == "Late") & DAYS_LATE <= 2]),
  UnitsArriveLessThan5 = sum(Units[(Lateness == "OnTime" |
  Lateness == "Late") & DAYS_LATE <= 5])
  ) %>%
  droplevels()
```

Output Files

Lastly, Let's output the data frames we created to csv so that we can use the presentation tool of our choice.

```
# Output Tables to .csv ----
write_csv(OTS_by_Category[, c(1:4, 6:10, 5, 11:15)],
          path = paste(SOT_OTS_directory,  'OTS_by_Category.csv', sep = '/'))

write_csv(OTS_by_Vendor,
          path = paste(SOT_OTS_directory,  'OTS_by_Vendor.csv', sep = '/'))

write_csv(SOT_by_Category,
          path = paste(SOT_OTS_directory,  'SOT_by_Category.csv', sep = '/'))
write_csv(SOT_by_Vendor,
          path = paste(SOT_OTS_directory,  'SOT_by_Vendor.csv', sep = '/'))

# YTD Masters
write_csv(SOT_Master, path = paste(
  SOT_OTS_directory,
  paste('SOT_Master_WK', EOW, '_YTD.csv', sep = ""),
  sep = '/'))

write_csv(OTS_Master, path = paste(
  SOT_OTS_directory,
  paste('OTS_Master_WK', EOW, '_YTD.csv', sep = ""),
  sep = '/'))

# 7 day Masters
write_csv(subset(SOT_Master, ShipCancelWeek == EOW), path = paste(
  SOT_OTS_directory,
  paste('SOT_Master_WK', EOW, '.csv', sep = ""),
  sep = '/'))

write_csv(subset(OTS_Master, Week == EOW), path = paste(
  SOT_OTS_directory,
  paste('OTS_Master_WK', EOW, '.csv', sep = ""),
  sep = '/'))
```