This program is written in R programming language version 3.1.1 installed on a Linux server. "R is a free software environment for statistical computing and graphics" with
no guarantees. R compiles and runs on a wide variety of UNIX platforms, Windows and MacOS." To download a free copy of R visit "http://www.r-project.org/".
In addition to base R, the following R packages were used in this analysis:
package "foreach" version 1.4.0
package "data.table" version 1.9.4
package "reshape2" version 1.2.1
package "XLConnect" version 0.2-10
package "zoo" version 1.7-7

This program will download from the internet and install the latest version of the above packages If they are not installed in your R environment. It is necessary to
have internet connection to download these packages.

If for any reason this program fails to run, please make sure that the above packages are installed, check the verion of the packages and
make sure the functions called in this program are still in use and are compatible with the Operating System you are using.

A step-by-step description is provided throughout this code.

##

Load Necessary Packages for this analysis

if (!(require(foreach))) install.packages ("foreach")
if (!(require(data.table))) install.packages ("data.table")
if (!(require(zoo))) install.packages ("zoo")

You will need to download Fannie Mae's Single-Family Loan Performance Data from Fannie Mae's website at https://loanperformancedata.fanniemae.com/lppub/index.html.
For more detail please refer to the accompanied presentation. After downloading the files you will need to unzip the files.
Though read.table function in R can read zipped files, we have used the "fread" function from data.table package
to read these files for efficiency and speed. Unfortunately, fread cannot read zipped files.
While this program will run with any number of pairs of files, we encourage users to download the entire set of Acquisition and Performance
files. The naming of the files should remain the same after download and unzipping process so that the files are saved in order.
This program will process the first Acquisition file and then the first Performance file, merge them together,
and then repeat that process for all matching files.

# You will need the path to where you have saved the downloaded files, please copy and paste or type the path below.
fileslocation<- "/<INSERT FILE(S) LOCATION HERE>/"

# Check the number of files downloaded (should be even, equal number of Acquisition and Performance Files).
numberoffiles<-length(list.files(fileslocation, pattern = glob2rx("*txt"), full.names=TRUE))

The "foreach" package contructs a loop so that R can iterate through all pairs of related Acquisition and Performance files.

```
# Calculate the number of iterations/cores in parallel processing allowing each pair to be processed simultaneously.
numberofloops<-(numberoffiles/2)

# Create function to handle missing Current UPBs in the last record by setting them to the record prior
na.lomf <- function(x) {

  na.lomf.0 <- function(x) {
    non.na.idx <- which(!is.na(x))
    if (is.na(x[1L])) {
      non.na.idx <- c(1L, non.na.idx)
    }
    rep.int(x[non.na.idx], diff(c(non.na.idx, length(x) + 1L)))
  }

  dim.len <- length(dim(x))

  if (dim.len == 0L) {
    na.lomf.0(x)
  } else {
    apply(x, dim.len, na.lomf.0)
  }
}

#########################################################################
# Start of Part 1; Data Preperation Step
#########################################################################

# After defining the Acquisition and Performance variables and their classes, the files are read into R and then data
manipulation is carried out.
# Acquisition and Performance files (from one or many quarters) will be merged into an R dataframe called
"Combined_Data."
Combined_Data <- foreach(k=1:numberofloops, .inorder=FALSE,
          .packages=c("data.table", "zoo")) %do% {

# Define Acquisition variables and classes, and read the files into R.
Acquisitions <- list.files(fileslocation, pattern = glob2rx("*Acquisition*txt"), full.names=TRUE)

Acquisitions_Variables = c("LOAN_ID", "ORIG_CHN", "Seller.Name", "ORIG_RT", "ORIG_AMT", "ORIG_TRM",
"ORIG_DTE"
                           ,"FRST_DTE", "OLTV", "OCLTV", "NUM_BO", "DTI", "CSCORE_B", "FTHB_FLG",
"PURPOSE", "PROP_TYP"
                           ,"NUM_UNIT", "OCC_STAT", "STATE", "ZIP_3", "MI_PCT", "Product.Type", "CSCORE_C")

Acquisition_ColClasses = c("character", "character", "character", "numeric", "numeric", "integer", "character",
"character", "numeric",
                           "numeric", "character", "numeric", "numeric", "character", "character", "character", "character",
"character",
                           "character", "character", "numeric", "character", "numeric")

Data_A<- fread(Acquisitions[k], sep = "|", colClasses = Acquisition_ColClasses, showProgress=FALSE)
setnames(Data_A, Acquisitions_Variables)
setkey(Data_A, "LOAN_ID")

# Delete unnecessary Acquisition variables.
```

```
Data_A[,c("Seller.Name","Product.Type"):=NULL]

# Obtain the Minimum Fico Score of the Borrower and Co-Borrower, Calculate House Price, and Replace Missing
OCLTV values with OLTV values where available
Data_A[, c("CSCORE_MN", "ORIG_VAL", "OCLTV"):= list(pmin(CSCORE_B,CSCORE_C, na.rm = TRUE),
                                    (ORIG_AMT/(OLTV/100)),
                                    ifelse(is.na(OCLTV), OLTV, OCLTV))]

# Remove not-needed Acquisition data from R environment.
rm('Acquisitions_Variables', 'Acquisition_ColClasses')

# Define Performance variables and classes, and read the files into R.
Performance_Variables = c("LOAN_ID", "Monthly.Rpt.Prd", "Servicer.Name", "LAST_RT", "LAST_UPB",
"Loan.Age", "Months.To.Legal.Mat"
                  , "Adj.Month.To.Mat", "Maturity.Date", "MSA", "Delq.Status", "MOD_FLAG", "Zero.Bal.Code",
                  "ZB_DTE", "LPI_DTE", "FCC_DTE","DISP_DT", "FCC_COST", "PP_COST", "AR_COST",
"IE_COST", "TAX_COST", "NS_PROCS",
                 "CE_PROCS", "RMW_PROCS", "O_PROCS")

Performance_ColClasses = c("character", "character", "character", "numeric", "numeric", "numeric", "numeric",
"numeric", "character",
                   "character", "character", "character", "character", "character", "character", "character", "character",
                   "numeric", "numeric", "numeric", "numeric", "numeric", "numeric", "numeric", "numeric", "numeric")

Performance <- list.files(fileslocation, pattern = glob2rx("*Performance*txt"), full.names=TRUE)

# Read and Process Performance data
Data_P = fread(Performance[k], sep = "|", colClasses = Performance_ColClasses, showProgress=FALSE)
setnames(Data_P, Performance_Variables)

# Apply function to backfill missing current UPBs
Data_P$LAST_UPB <- na.lomf(Data_P$LAST_UPB)

# Convert character variables to Date type
Data_P$Monthly.Rpt.Prd<-as.Date(Data_P$Monthly.Rpt.Prd, "%m/%d/%Y")
Data_P$DISP_DT<-as.Date(Data_P$DISP_DT, "%m/%d/%Y")
Data_P$FCC_DTE<-as.Date(Data_P$FCC_DTE, "%m/%d/%Y")

# Sort data by Loan ID and Monthly Reporting Period
setorderv(Data_P, c("LOAN_ID", "Monthly.Rpt.Prd"))
setkey(Data_P, "LOAN_ID")

# Standardize Delinquency Status Codes
Data_P$Delq.Status<-as.numeric(ifelse(Data_P$Delq.Status=="X", "999", Data_P$Delq.Status))

# Count the number of months a loan is active
Data_P[,Count:=1:.N, by="LOAN_ID"]

# Obtain the date of the first time each loan was modified
FMOD_DTE = Data_P[, .SD[MOD_FLAG =="Y"][,FMOD_DTE:=Monthly.Rpt.Prd]][, .SD[1], by =
"LOAN_ID"][,c("LOAN_ID", "FMOD_DTE"), with = FALSE, drop = FALSE]

# Obtain the date and UPB of each loan's first credit event (i.e. 180 days SDQ, or Foreclosure or Default)
First_CE = Data_P[, .SD[Zero.Bal.Code =="03" | Zero.Bal.Code =="09"
```

```
                | (Delq.Status<999 & Delq.Status>= 6)][,c("FCE_DTE", "FCE_UPB", "SPDelq1", "CountFC")
                                        := list(Monthly.Rpt.Prd, LAST_UPB, Delq.Status, Count)]][, .SD[1], by =
"LOAN_ID"][,c("LOAN_ID", "SPDelq1", "FCE_DTE", "FCE_UPB", "CountFC"), with = FALSE, drop = FALSE]

# Obtain the date and UPB of each loan becoming 180 days delinquent
First_D180 = Data_P[, .SD[Delq.Status<999 & Delq.Status >=6][,c("F180_DTE", "F180_UPB", "SPDelq2",
"CountF1"):=
                                        list(Monthly.Rpt.Prd, LAST_UPB, Delq.Status, Count)]][, .SD[1], by =
"LOAN_ID"][,c("LOAN_ID", "F180_DTE", "F180_UPB", "SPDelq2", "CountF1"), with = FALSE, drop = FALSE]

# Summarize Perfomance data by keeping only the last row of a loan's activity
Data_P<-Data_P[, .SD[.N], by ="LOAN_ID"]

# Define the last status of a loan and calculate the months between Last Paid Installment and Disposition date (for Lost
Interest calculation)
Data_P[, c("LAST_STAT", "lpi2disp"):=
    list(ifelse(Zero.Bal.Code=='01','P',ifelse(Zero.Bal.Code=='03','S', ifelse(Zero.Bal.Code=='06', 'R',
ifelse(Zero.Bal.Code=='09', 'F', ifelse(Delq.Status=='999','X',ifelse(Delq.Status >9, '9', ifelse(Delq.Status==0, 'C',
as.character(Delq.Status)))))))),
        ifelse(Data_P$LPI_DTE!="" & !(is.na(Data_P$DISP_DT)),as.numeric((year(DISP_DT)-
year(as.yearmon(LPI_DTE, "%m/%d/%Y")))*12+month(DISP_DT)-month(as.yearmon(LPI_DTE, "%m/%d/%Y"))),
0))]

# Calculate Interest Cost, total expenses and total proceeds
Data_P[, c("INT_COST","total_expense", "total_proceeds") :=
    list(ifelse(((LAST_STAT =="F" | LAST_STAT =="S")& !is.na(DISP_DT)),LAST_UPB*(((LAST_RT/100) -
.0035)/12)*lpi2disp,0),
        ifelse((LAST_STAT =="F" | LAST_STAT =="S"),rowSums(Data_P[,
list(FCC_COST,PP_COST,AR_COST,TAX_COST,IE_COST)], na.rm = TRUE),0),
        ifelse((LAST_STAT =="F" | LAST_STAT =="S"),(-1*rowSums(Data_P[,
list(NS_PROCS,CE_PROCS,RMW_PROCS,O_PROCS)], na.rm = TRUE)),0))]

# Calculate Net Loss, Net Severity, Total Costs, Total Proceeds, and Total Liquidation Expenses.  Define Last Date
variable.
Data_P[,c("NET_LOSS","NET_SEV", "Total_Cost", "Tot_Procs", "Tot_Liq_Ex", "LAST_DTE"):=
          list(ifelse(((LAST_STAT =="F" | LAST_STAT =="S")& !is.na(DISP_DT)),rowSums(Data_P[,
list(LAST_UPB,INT_COST,total_expense,total_proceeds)], na.rm=TRUE),0),
              ifelse(((LAST_STAT =="F" | LAST_STAT =="S")& !is.na(DISP_DT)),(rowSums(Data_P[,
list(LAST_UPB,INT_COST,total_expense,total_proceeds)], na.rm=TRUE)/LAST_UPB),0),
              ifelse((LAST_STAT =="F" | LAST_STAT =="S"),rowSums(Data_P[, list(LAST_UPB,
INT_COST,FCC_COST,PP_COST, AR_COST, IE_COST, TAX_COST)], na.rm = TRUE),0),
              ifelse((LAST_STAT =="F" | LAST_STAT =="S"),rowSums(Data_P[, list(NS_PROCS, CE_PROCS,
RMW_PROCS, O_PROCS)], na.rm = TRUE),0),
              ifelse((LAST_STAT =="F" | LAST_STAT =="S"),rowSums(Data_P[, list(FCC_COST, PP_COST,
AR_COST, IE_COST, TAX_COST)], na.rm = TRUE),0),
              as.Date(ifelse(!(is.na(Data_P$DISP_DT)), Data_P$DISP_DT, ifelse(!(is.na(Data_P$FCC_DTE)),
Data_P$FCC_DTE, Data_P$Monthly.Rpt.Prd))))]


# Merge new fields with full performance dataset to capture information on First Modification, First Credit Event, and
First Default.
Data_P[FMOD_DTE, FMOD_DTE:=i.FMOD_DTE]
Data_P[First_CE, c("FCE_DTE", "FCE_UPB", "SPDelq1", "CountFC"):=list(i.FCE_DTE, i.FCE_UPB, i.SPDelq1,
i.CountFC)]
```

```
Data_P[First_D180, c("F180_DTE", "F180_UPB", "SPDelq2", "CountF1"):=list(i.F180_DTE, i.F180_UPB, i.SPDelq2, i.CountF1)]

# Delete Performance variables that are not needed.
Data_P[, c("Count", "Monthly.Rpt.Prd", "ZB_DTE", "Servicer.Name", "Loan.Age", "Months.To.Legal.Mat", "Adj.Month.To.Mat", "Maturity.Date", "Delq.Status","total_expense", "total_proceeds", "lpi2disp"):=NULL]

# Remove not-needed data from R environment.
rm("First_D180", "First_CE", "FMOD_DTE", "Performance_Variables", "Performance_ColClasses")


# Merge together full Acquisition and Performance files.
Combined_Data = as.data.table(merge(Data_A, Data_P, by.x = "LOAN_ID", by.y = "LOAN_ID", all = TRUE))

# Create Vintage Year & Activity Year Attributes, set missing F180_UPB and FCE_UPB equal to ORIG_AMT if the loan goes to delinquency during the
# first six month of loan activity.
Combined_Data[,c("VinYr", "ActYr", "DispYr", "F180_UPB", "FCE_UPB") :=list(format(as.yearmon(ORIG_DTE, format="%m/%Y"), "%Y"),
                                                format(as.yearmon(LAST_DTE, format="%m/%Y"), "%Y"),
                                                format(as.yearmon(DISP_DT, format="%m/%Y"), "%Y"),
                                                ifelse((SPDelq2==6 & is.na(F180_UPB) & CountF1<=6), ORIG_AMT,
                                                    ifelse(!(is.na(F180_UPB)),F180_UPB ,0)),
                                                ifelse((SPDelq1==6 & CountFC <=6 & is.na(FCE_UPB)), ORIG_AMT,
                                                    ifelse(!(is.na(FCE_UPB)),FCE_UPB ,0)))]

Combined_Data[,c("SPDelq1","SPDelq2", "CountF1", "CountFC"):=NULL]

#rm(list= ls()[!(ls() %in% c('Combined_Data'))])

return(Combined_Data)

}

Combined_Data<-rbindlist(Combined_Data, fill=TRUE)

# Save a Copy to disk or write a .txt file.
save(Combined_Data, file="FNMA_Performance_Data.Rda")

# Remove all objects created besides the final data set.
rm(list= ls()[!(ls() %in% c('Combined_Data'))])

#########################################################################
# End of Part 1; Data Preperation Step
#########################################################################




#########################################################################
# Start of Part 2; Summary Statistics Step
#########################################################################
#########################################################################
# Below various summary statistics are calculated and outputed to an .XLSX file. We use the XLConnect package to
```

```
write the summary statistics to the .xlsx file.
# The file will be written to current working directory, if you want to change the location of the file please specify the location followed by file name.
# to get the current working directory type getwd() in your R console. You can also change the format of the file to an xls file by changing the file extension.
# Summary statistics will be outputted as separate tabs in the xls or xlsx file.
######################################################################

if (!(require(XLConnect))) install.packages ("XLConnect")
if (!(require(reshape2))) install.packages ("reshape2")

#Turn off scientific notation to prevent UPB round
options(scipen=999)

# Create the output file, change the name and location of the file below
Charts<-loadWorkbook("Statistics.xlsx", create = TRUE)

# Create buckets for continuous attributes, Risk Flag, and group number of borrowers
Combined_Data[,c("FicoBkt")
        :=list(as.character(cut(CSCORE_MN, breaks = c(-Inf, 0, 620, 660, 700, 740, 780, Inf),
                        labels = c('NA','[0-620)', '[620-660)', '[660-700)', '[700-740)', '[740-780)', '[780+)'),
                        right = FALSE, ordered = TRUE)))]

# Create 'Missing' buckets for continuous attributes
Combined_Data$FicoBkt[is.na(Combined_Data$FicoBkt)] <- 'MissingFICO'


# The following section will produce tables that will help users tie out their loan counts to the loan counts in the webiner
# Loan counts cut by origination vintage and purpose
Vint.REFI.Counts<-as.data.frame(addmargins(xtabs(~PURPOSE+VinYr, data=Combined_Data)))
Vint.REFI.Counts<-dcast(Vint.REFI.Counts,PURPOSE~VinYr,value.var = "Freq")
createSheet(Charts, name = "Vint.REFI.Counts")
writeWorksheet(Charts, Vint.REFI.Counts, sheet = "Vint.REFI.Counts", startRow = 1, startCol = 1)

# Loan counts cut by origination vintage and occupancy
Vint.OCC.Counts<-as.data.frame(addmargins(xtabs(~OCC_STAT+VinYr, data=Combined_Data)))
Vint.OCC.Counts<-dcast(Vint.OCC.Counts,OCC_STAT~VinYr,value.var = "Freq")
createSheet(Charts, name = "Vint.OCC.Counts")
writeWorksheet(Charts, Vint.OCC.Counts, sheet = "Vint.OCC.Counts", startRow = 1, startCol = 1)

# Loan counts cut by last_status
Vint.LAST_STAT.Counts<-as.data.frame(addmargins(xtabs(~LAST_STAT, data=Combined_Data)))
createSheet(Charts, name = "Vint.LAST_STAT.Counts")
writeWorksheet(Charts, Vint.LAST_STAT.Counts, sheet = "Vint.LAST_STAT.Counts", startRow = 1, startCol = 1)

#Summary Stats for Fico, Original Amount and OLTV
Summary<-as.data.frame(unstack(as.data.frame(summary(Combined_Data[, list(CSCORE_MN, OLTV,
ORIG_AMT)])), Freq~Var2))
names(Summary)<-c("CSCORE_MN", "OLTV", "ORIG_AMT")
createSheet(Charts, name = "Summary Stats")
writeWorksheet(Charts, Summary, sheet = "Summary Stats", startRow = 1, startCol = 1)

# Loan counts by FICO bucket and origination vintage
```

```
Vint.Fico.Counts<-as.data.frame(addmargins(xtabs(~FicoBkt+VinYr, data=Combined_Data)))
Vint.Fico.Counts<-dcast(Vint.Fico.Counts,FicoBkt~VinYr,value.var = "Freq")
createSheet(Charts, name = "Vint.Fico.Counts")
writeWorksheet(Charts, Vint.Fico.Counts, sheet = "Vint.Fico.Counts", startRow = 1, startCol = 1)


# Acquisition Summary Statistics by Vintage
Aqsn.Stat1<-setorder(Combined_Data[, list(
  "Loan Count"= .N,
  "Total Orig. UPB"= sum(ORIG_AMT, na.rm = TRUE),
  "Avg. Orig UPB($)"= round(mean(ORIG_AMT, na.rm = TRUE)),
  "Borrower Credit Score"= round(weighted.mean(CSCORE_B, ORIG_AMT, na.rm=TRUE),0),
  "Co-Borrower Credit Score"= round(weighted.mean(CSCORE_C, ORIG_AMT, na.rm=TRUE),0),
  "LTV Ratio"= sprintf("%.2f",weighted.mean(OLTV, ORIG_AMT, na.rm=TRUE)),
  "CLTV Ratio"= sprintf("%.2f",weighted.mean(OCLTV, ORIG_AMT, na.rm=TRUE)),
  "DTI"= sprintf("%.2f",weighted.mean(DTI, ORIG_AMT, na.rm=TRUE)),
  "Note Rate"= sprintf("%.2f",weighted.mean(ORIG_RT, ORIG_AMT, na.rm=TRUE))), by=list(Vintage=VinYr)],
"Vintage")

# Acquisition Stat Totals
Aqsn.Stat2<-Combined_Data[, list(
  Vintage= "Total",
  "Loan Count"= .N,
  "Total Orig. UPB"= sum(ORIG_AMT, na.rm = TRUE),
  "Avg. Orig UPB($)"= round(mean(ORIG_AMT, na.rm = TRUE)),
  "Borrower Credit Score"= round(weighted.mean(CSCORE_B, ORIG_AMT, na.rm=TRUE),0),
  "Co-Borrower Credit Score"= round(weighted.mean(CSCORE_C, ORIG_AMT, na.rm=TRUE),0),
  "LTV Ratio"= sprintf("%.2f",weighted.mean(OLTV, ORIG_AMT, na.rm=TRUE)),
  "CLTV Ratio"= sprintf("%.2f",weighted.mean(OCLTV, ORIG_AMT, na.rm=TRUE)),
  "DTI"= sprintf("%.2f",weighted.mean(DTI, ORIG_AMT, na.rm=TRUE)),
  "Note Rate"= sprintf("%.2f",weighted.mean(ORIG_RT, ORIG_AMT, na.rm=TRUE)))]

# Merge Totals with breakout by Vintage for Full Acquisition Statistics Table
Aqsn.Stat<-rbind(Aqsn.Stat1,Aqsn.Stat2)
createSheet(Charts, name = "Aqsn.Stat")
writeWorksheet(Charts, Aqsn.Stat, sheet = "Aqsn.Stat", startRow = 1, startCol = 1)

rm(Aqsn.Stat1, Aqsn.Stat2)

# Performance Loan Counts by Vintage
Perf.Stat1<-setorder(Combined_Data[, list(
  "Loan Count"= .N,
  "Total Orig. UPB"= sum(ORIG_AMT, na.rm = TRUE),
  "Loan Count (Active)"= sum(ifelse(LAST_STAT %chin% c("C", "1", "2", "3", "4", "5", "6", "7", "8", "9"), 1, 0),
na.rm=TRUE),
  "Active UPB"= sum(ifelse(LAST_STAT %chin% c("C", "1", "2", "3", "4", "5", "6", "7", "8", "9"), LAST_UPB, 0),
na.rm=TRUE),
  "Prepaid"= sum(ifelse(LAST_STAT =="P", 1, 0), na.rm=TRUE),
  "Repurchased"= sum(ifelse(LAST_STAT =="R", 1, 0), na.rm=TRUE),
  "Alternative Disposition"= sum(ifelse(LAST_STAT %chin% c("S","999"), 1, 0), na.rm=TRUE),
  "REO Disposition"= sum(ifelse(LAST_STAT=="F", 1, 0), na.rm=TRUE),
  "Modified"= sum(ifelse(MOD_FLAG =="Y", 1, 0), na.rm=TRUE),
  "Default UPB"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),LAST_UPB, 0) ,
na.rm=TRUE),
```

```
  "Net Loss Rate"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),NET_LOSS, 0), na.rm =
TRUE)/sum(ORIG_AMT, na.rm = TRUE))
  , by=list(Vintage=VinYr)], "Vintage")

# Performance Loan Count Totals
Perf.Stat2<-Combined_Data[, list(
  Vintage= "Total",
  "Loan Count"= .N,
  "Total Orig. UPB"= sum(ORIG_AMT, na.rm = TRUE),
  "Loan Count (Active)"= sum(ifelse(LAST_STAT %chin% c("C", "1", "2", "3", "4", "5", "6", "7", "8", "9"), 1, 0),
na.rm=TRUE),
  "Active UPB"= sum(ifelse(LAST_STAT %chin% c("C", "1", "2", "3", "4", "5", "6", "7", "8", "9"), LAST_UPB, 0),
na.rm=TRUE),
  "Prepaid"= sum(ifelse(LAST_STAT =="P", 1, 0), na.rm=TRUE),
  "Repurchased"= sum(ifelse(LAST_STAT =="R", 1, 0), na.rm=TRUE),
  "Alternative Disposition"= sum(ifelse(LAST_STAT %chin% c("S","999"), 1, 0), na.rm=TRUE),
  "REO Disposition"= sum(ifelse(LAST_STAT=="F", 1, 0), na.rm=TRUE),
  "Modified"= sum(ifelse(MOD_FLAG =="Y", 1, 0), na.rm=TRUE),
  "Default UPB"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),LAST_UPB, 0) ,
na.rm=TRUE),
  "Net Loss Rate"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),NET_LOSS, 0), na.rm =
TRUE)/sum(ORIG_AMT, na.rm = TRUE))]

# Merge Totals with breakout by Vintage for Full Performance Statistics Table of Loan Counts
Perf.Stat<-rbind(Perf.Stat1, Perf.Stat2)
createSheet(Charts, name = "Perf.Stat.Counts")
writeWorksheet(Charts, Perf.Stat, sheet = "Perf.Stat.Counts", startRow = 1, startCol = 1)

rm(Perf.Stat1, Perf.Stat2)

# Performance UPB broken out by Vintage
Perf.Stat.Sums1<-setorder(Combined_Data[, list(
  "Loan Count"= .N,
  "Total Orig. UPB"= sum(ORIG_AMT, na.rm = TRUE),
  "Active UPB"= sum(ifelse(LAST_STAT %chin% c("C", "1", "2", "3", "4", "5", "6", "7", "8", "9"), LAST_UPB, 0),
na.rm=TRUE),
  "Prepaid UPB"= sum(ifelse(LAST_STAT =="P", LAST_UPB, 0), na.rm=TRUE),
  "REO Disposition UPB"= sum(ifelse(LAST_STAT=="F", LAST_UPB, 0), na.rm=TRUE),
  "Alternative Disposition"= sum(ifelse(LAST_STAT %chin% c("S","999"), LAST_UPB, 0), na.rm=TRUE),
  "Repurchased UPB"= sum(ifelse(LAST_STAT =="R", LAST_UPB, 0), na.rm=TRUE),
  "Modified UPB"= sum(ifelse(MOD_FLAG =="Y", LAST_UPB, 0), na.rm=TRUE),
  "Default UPB"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),LAST_UPB, 0) ,
na.rm=TRUE),
  "Net Loss Rate"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),NET_LOSS, 0), na.rm =
TRUE)/sum(ORIG_AMT, na.rm = TRUE))
  , by=list(Vintage=VinYr)], "Vintage")

# Performance UPB Totals
Perf.Stat.Sums2<-Combined_Data[, list(
  Vintage= "Total",
  "Loan Count"= .N,
  "Total Orig. UPB"= sum(ORIG_AMT, na.rm = TRUE),
  "Active UPB"= sum(ifelse(LAST_STAT %chin% c("C", "1", "2", "3", "4", "5", "6", "7", "8", "9"), LAST_UPB, 0),
na.rm=TRUE),
```

```
  "Prepaid UPB"= sum(ifelse(LAST_STAT =="P", LAST_UPB, 0), na.rm=TRUE),
  "REO Disposition UPB"= sum(ifelse(LAST_STAT=="F", LAST_UPB, 0), na.rm=TRUE),
  "Alternative Disposition"= sum(ifelse(LAST_STAT %chin% c("S","999"), LAST_UPB, 0), na.rm=TRUE),
  "Repurchased UPB"= sum(ifelse(LAST_STAT =="R", LAST_UPB, 0), na.rm=TRUE),
  "Modified UPB"= sum(ifelse(MOD_FLAG =="Y", LAST_UPB, 0), na.rm=TRUE),
  "Default UPB"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),LAST_UPB, 0) ,
na.rm=TRUE),
  "Net Loss Rate"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),NET_LOSS, 0), na.rm =
TRUE)/sum(ORIG_AMT, na.rm = TRUE))]

# Merge Totals with breakout by Vintage for Full Performance Statistics Table of UPB amounts
Perf.Stat.Sums<-rbind(Perf.Stat.Sums1, Perf.Stat.Sums2)
createSheet(Charts, name = "Perf.Stat.Sums")
writeWorksheet(Charts, Perf.Stat.Sums, sheet = "Perf.Stat.Sums", startRow = 1, startCol = 1)

rm(Perf.Stat.Sums1, Perf.Stat.Sums2)

# Historical Net Loss Statistics by Vintage
HistNetLoss1a<-setorder(Combined_Data[,list(
  "Loan Count"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), 1, 0)),
  "UPB for Liquiditions"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm
= TRUE),
  "Default UPB % of Orig. UPB"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm=TRUE)/sum(ORIG_AMT, na.rm = TRUE))*100),
  "Interest on Delinquent Loans"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), INT_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Total Liquidition Exp."= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
Tot_Liq_Ex, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Foreclosure Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
FCC_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Prop.Pres. Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
PP_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Asset Recovery Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
AR_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Miscellaneous Holding Expenses And Credits"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S")
& !(is.na(DISP_DT))), IE_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Associated Taxes"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
TAX_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Total Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), Total_Cost,
0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm =
TRUE))*100),
  "Sales Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
NS_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Credit Enhancement Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), CE_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
```

```
  "Repurchase/Make Whole Proceeds"=sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), RMW_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Other Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
O_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Total Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
Tot_Procs, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Severity"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), NET_LOSS,
0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm =
TRUE))*100),
  "Total Net Loss"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), NET_LOSS, 0), na.rm =
TRUE))
  , by=list(Vintage=VinYr)], "Vintage")

# Historical Loss Totals
HistNetLoss1b<-setorder(Combined_Data[,list(
  "Vintage"= "Total",
  "Loan Count"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), 1, 0)),
  "UPB for Liquiditions"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm
= TRUE),
  "Default UPB % of Orig. UPB"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm=TRUE)/sum(ORIG_AMT, na.rm = TRUE))*100),
  "Interest on Delinquent Loans"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), INT_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Total Liquidition Exp."= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
Tot_Liq_Ex, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Foreclosure Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
FCC_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Prop.Pres. Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
PP_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Asset Recovery Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
AR_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Miscellaneous Holding Expenses And Credits"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S")
& !(is.na(DISP_DT))), IE_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Associated Taxes"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
TAX_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Total Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), Total_Cost,
0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm =
TRUE))*100),
  "Sales Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
NS_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Credit Enhancement Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), CE_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
```

```
  "Repurchase/Make Whole Proceeds"=sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), RMW_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Other Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
O_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Total Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
Tot_Procs, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Severity"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), NET_LOSS,
0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm =
TRUE))*100),
  "Total Net Loss"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), NET_LOSS, 0), na.rm =
TRUE))], "Vintage")

# Merge Totals with breakout by Vintage for Full Historical Net Loss Table
HistNetLosso<-rbind(HistNetLoss1a,HistNetLoss1b)
HistNetLoss2<-as.data.frame(t(as.data.frame(HistNetLosso)))
colnames(HistNetLoss2)<-unique(HistNetLosso$Vintage)
HistNetLoss2<-HistNetLoss2[2:nrow(HistNetLoss2),]

createSheet(Charts, name = "Orig.Loss.Stat")
writeWorksheet(Charts, HistNetLoss2, sheet = "Orig.Loss.Stat", startRow = 1, startCol = 1, rownames="Row Names")

# Historical Net Loss Statistics by Disposition Year
HistNetLoss1c<-setorder(Combined_Data[,list(
  "Loan Count"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), 1, 0)),
  "UPB for Liquiditions"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm
= TRUE),
  "Default UPB % of Orig. UPB"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm=TRUE)/sum(ORIG_AMT, na.rm = TRUE))*100),
  "Interest on Delinquent Loans"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), INT_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Total Liquidition Exp."= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
Tot_Liq_Ex, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Foreclosure Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
FCC_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Prop.Pres. Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
PP_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Asset Recovery Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
AR_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Miscellaneous Holding Expenses And Credits"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S")
& !(is.na(DISP_DT))), IE_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Associated Taxes"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
TAX_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Total Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), Total_Cost,
0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm =
```

```
TRUE))*100),
  "Sales Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
NS_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Credit Enhancement Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), CE_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Repurchase/Make Whole Proceeds"=sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), RMW_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Other Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
O_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Total Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
Tot_Procs, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Severity"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), NET_LOSS,
0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm =
TRUE))*100),
  "Total Net Loss"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), NET_LOSS, 0), na.rm =
TRUE))
  , by=list(Disposition=DispYr)], "Disposition")

# Historical Loss Totals
HistNetLoss1d<-setorder(Combined_Data[,list(
  "Disposition"= "Total",
  "Loan Count"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), 1, 0)),
  "UPB for Liquiditions"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm
= TRUE),
  "Default UPB % of Orig. UPB"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm=TRUE)/sum(ORIG_AMT, na.rm = TRUE))*100),
  "Interest on Delinquent Loans"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), INT_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Total Liquidition Exp."= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
Tot_Liq_Ex, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Foreclosure Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
FCC_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Prop.Pres. Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
PP_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Asset Recovery Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
AR_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Miscellaneous Holding Expenses And Credits"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S")
& !(is.na(DISP_DT))), IE_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Associated Taxes"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
TAX_COST, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Total Costs"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), Total_Cost,
0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm =
```

```
TRUE))*100),
  "Sales Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
NS_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB,
0), na.rm = TRUE))*100),
  "Credit Enhancement Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), CE_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Repurchase/Make Whole Proceeds"=sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), RMW_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") &
!(is.na(DISP_DT))), LAST_UPB, 0), na.rm = TRUE))*100),
  "Other Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
O_PROCS, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Total Proceeds"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))),
Tot_Procs, 0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0),
na.rm = TRUE))*100),
  "Severity"= sprintf("%.2f%%", (sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), NET_LOSS,
0), na.rm = TRUE)/sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), LAST_UPB, 0), na.rm =
TRUE))*100),
  "Total Net Loss"= sum(ifelse((LAST_STAT %chin% c("F", "S") & !(is.na(DISP_DT))), NET_LOSS, 0), na.rm =
TRUE))], "Disposition")

# Merge Totals with breakout by Vintage for Full Historical Net Loss Table
HistNetLossd<-rbind(HistNetLoss1c,HistNetLoss1d)
HistNetLoss3<-as.data.frame(t(as.data.frame(HistNetLossd)))
colnames(HistNetLoss3)<-unique(HistNetLossd$Disposition)
HistNetLoss3<-HistNetLoss3[2:nrow(HistNetLoss3),]

createSheet(Charts, name = "Disp.Loss.Stat")
writeWorksheet(Charts, HistNetLoss3, sheet = "Disp.Loss.Stat", startRow = 1, startCol = 1, rownames="Row Names")

# Save the .xlsx document of all tables
saveWorkbook(Charts)

# Removing full dataset from R Environment
rm(list= ls()[!(ls() %in% c('Combined_Data'))])

#########################################################################
# End of Part 2; Summary Statistics Step
#########################################################################
```