```
libname temp '/<FILE PATH FOR WORKING DIRECTORY>/';
libname myfolder '/<FILE PATH FOR FINAL DATASET>/';

options errors= 2;
options nocenter ps=52 errors=1 compress=yes obs=max;

** directory with performance and acquisition .txt files **;
%let directory = /<INSERT FILE(S) LOCATION HERE>/;

*******************************************************************;
** file headers - variables beginning in 'x' will be reformatted **;
*******************************************************************;
%let acq_head =
loan_id :$12.
orig_chn :$1.
seller :$80.
orig_rt
orig_amt
orig_trm
x_orig_date :$7.
x_first_pay :$7.
oltv
ocltv
num_bo
dti
cscore_b
fthb_flg :$1.
purpose :$1.
prop_typ :$2.
num_unit
occ_stat :$1.
state :$2.
zip_3 :$5.
mi_pct
x_prod_type :$3.
cscore_c
;

%let perf_head =
loan_id :$12.
x_period :$10.
y_servicer :$80.
y_curr_rte
y_act_upb
x_loan_age
x_rem_mths
x_adj_rem_months
x_maturity_date :$7.
msa :$5.
x_dlq_status :$3.
y_mod_ind :$1.
z_zb_code :$2.
x_zb_date :$7.
```

```
x_lpi_dte :$10.
x_fcc_dte :$10.
x_disp_dte :$10.
fcc_cost
pp_cost
ar_cost
ie_cost
tax_cost
ns_procs
ce_procs
rmw_procs
o_procs
;

%macro lppub (quarter);
****************************************;
** upload and format acquisition files **;
****************************************;

data acq (drop = x_orig_date x_first_pay x_prod_type);
  infile "&directory.Acquisition_&quarter..txt" delimiter = '|' missover dsd lrecl=32767;
  input &acq_head;

   *converting date formats from mm/yyyy to sas date;
   format orig_dte frst_pay mmddyy8.;
   orig_dte = mdy(input(substr(x_orig_date,1,2),2.),1,input(substr(x_orig_date,4,4),4.));
   frst_pay = mdy(input(substr(x_first_pay,1,2),2.),1,input(substr(x_first_pay,4,4),4.));

run;

proc sort data = acq; by loan_id; run;

*************************************;
** upload and format activity files **;
*************************************;

data temp (drop = x_zb_date x_period x_maturity_date x_adj_rem_months x_rem_mths x_loan_age
                 x_dlq_status x_lpi_dte x_fcc_dte x_disp_dte);
  infile "&directory.Performance_&quarter..txt" delimiter = '|' missover dsd lrecl=32767;
  input &perf_head;

  *converting dates from mm/yyyy to sas date;
  format y_act_date z_zb_date lpi_dte fcc_dte disp_dte mmddyy8.;
  z_zb_date = mdy(input(substr(x_zb_date,1,2),2.),1,input(substr(x_zb_date,4,4),4.));

  *converting dates from mm/dd/yyyy to sas date;
  y_act_date = mdy(input(substr(x_period,1,2),2.),1,input(substr(x_period,7,4),4.));
  lpi_dte = mdy(input(substr(x_lpi_dte,1,2),2.),1,input(substr(x_lpi_dte,7,4),4.));
  fcc_dte = mdy(input(substr(x_fcc_dte,1,2),2.),1,input(substr(x_fcc_dte,7,4),4.));
  disp_dte = mdy(input(substr(x_disp_dte,1,2),2.),1,input(substr(x_disp_dte,7,4),4.));

  *to convert delinquency status from character to number, we set 'X' values to '999';
  if x_dlq_status = 'X' then y_dlq_stat = 999;
  else y_dlq_stat = x_dlq_status*1;
```

```
run;

*sorting loans by activity date to keep in chronological order;
proc sort data = temp; by loan_id y_act_date; run;

*****************************************************************************************************
**********************;
**---------------------- retaining elements from activity files required for static data set ----------------------**;
*****************************************************************************************************
**********************;

data act (drop = y_mod_ind y_servicer y_num_periods
       rename =( y_act_upb = last_upb
              y_dlq_stat = z_last_status
              y_curr_rte = last_rt
              y_act_date = last_activity_date ));
  set temp;
  by loan_id y_act_date;

  length servicer $80;
  retain servicer;
  retain y_num_periods f180_dte fce_dte f180_upb fce_upb mod_flag fmod_dte z_num_periods_180
z_num_periods_ce z_num_periods_lqd;
  format f180_dte fce_dte fmod_dte mmddyy8.;

  if first.loan_id then do;

    servicer = y_servicer;

    y_num_periods = 1;
    y_prev_upb = .;

   if 999 > y_dlq_stat >= 6 then f180_dte = y_act_date; else f180_dte = .;
   if 999 > y_dlq_stat >= 6 then f180_upb = y_act_upb;  else f180_upb= .;
   if 999 > y_dlq_stat >= 6 then z_num_periods_180 = y_num_periods; else z_num_periods_180 = .;

   if 999 > y_dlq_stat >= 6 and z_zb_code in ('03','09') then fce_dte = y_act_date; else fce_dte = .;
   if 999 > y_dlq_stat >= 6 and z_zb_code in ('03','09') then fce_upb = y_act_upb;  else fce_upb= .;
   if 999 > y_dlq_stat >= 6 and z_zb_code in ('03','09') then z_num_periods_ce = y_num_periods; else
z_num_periods_ce = .;

   if y_mod_ind = 'Y' then mod_flag = 1; else mod_flag = 0;
   if y_mod_ind = 'Y' then fmod_dte = y_act_date; else fmod_dte = .;

   *helps in capturing last upb of quick to liquidate loans*;
   if z_zb_code in ('01','03','06','09') then z_num_periods_lqd = y_num_periods; else z_num_periods_lqd = .;

  end;
  else do;

    *servicer field will capture the current servicer*;
    if y_servicer ne '' then servicer = y_servicer;
```

```
      y_num_periods = y_num_periods + 1;

      y_prev_upb = lag(y_act_upb);

      *capturing the last upb for zero balance loans;
      if y_act_upb <= 0 and z_zb_code in ('01','03','06','09') then y_act_upb = y_prev_upb;
      if z_zb_code in ('01','03','06','09') then z_num_periods_lqd = y_num_periods;

      *performance flags*;
      if 999 > y_dlq_stat >= 6 and f180_dte = . then do;
        f180_dte = y_act_date;
        f180_upb = y_act_upb;
        z_num_periods_180 = y_num_periods;
      end;
      if (999 > y_dlq_stat >= 6 or z_zb_code in ('03','09')) and fce_dte = . then do;
        fce_dte = y_act_date;
        fce_upb = y_act_upb;
        z_num_periods_ce = y_num_periods;
      end;

      if y_mod_ind = 'Y' and mod_flag = 0 then do;
        mod_flag = 1;
        fmod_dte = y_act_date;
      end;

   end;

   if last.loan_id;

run;



   **********************************;
   ** merge to create combined dataset **;
   **********************************;
data temp.comb_&quarter (drop = z_zb_code z_zb_date z_last_status z_num_periods_180 z_num_periods_ce
z_num_periods_lqd);
  merge acq   (in=a)
        act   (in=b);
  by loan_id;
  if a;

  ** correcting the null ce values on early dlq loans upb values **;
  if 0 < z_num_periods_180 <= 8 then f180_upb = orig_amt;
  if 0 < z_num_periods_ce  <= 8 then fce_upb  = orig_amt;
  if 0 < z_num_periods_lqd <= 8 and  last_upb <= 0 then last_upb = orig_amt;

  ** minimum credit score **;
  cscore_mn = min(cscore_b,cscore_c);

  **setting mipct equal to 0 when missing like freddie**;
  mi_pct = max(mi_pct,0);
  ** origination home value **;
  orig_val = orig_amt/(oltv/100);

  ** Set Missing OCLTVs to OLTV **;
  if OCLTV =. then OCLTV = OLTV;

  format last_dte date9.;
  if disp_dte ne . then last_dte = disp_dte;
  else if z_zb_date ne . then last_dte = z_zb_date;
  else last_dte = last_activity_date;

  *last status;
  length last_stat $1;
  if z_zb_code = '09'            then last_stat = "F";
  else if z_zb_code = '03'       then last_stat = "S";
  else if z_zb_code = '06'       then last_stat = "R";
  else if z_zb_code = '01'       then last_stat = "P";
  else if 999 > z_last_status > 9 then last_stat = "9";
  else if 9 >=  z_last_status > 0 then last_stat = put(z_last_status,1.);
  else if z_last_status = 0      then last_stat = "C";
  else                           last_stat = 'X';

  if last_stat in ("F","S") and disp_dte ne . then do;

    int_cost = intck("month",lpi_dte,last_dte)*(((last_rt/100)-0.0035)/12)*last_upb;
    net_loss = sum(last_upb,fcc_cost,pp_cost,ar_cost,ie_cost,tax_cost,int_cost,-1*ns_procs,-1*ce_procs,-1*rmw_procs,-
1*o_procs);
    net_sev = (net_loss/last_upb);

    if int_cost=. then int_cost=0;
    if net_loss=. then net_loss=0;
    if fcc_cost=. then fcc_cost=0;
    if pp_cost=. then pp_cost=0;
    if ar_cost=. then ar_cost=0;
    if ie_cost=. then ie_cost=0;
    if tax_cost=. then tax_cost=0;
    if ns_procs=. then ns_procs=0;
    if ce_procs=. then ce_procs=0;
    if rmw_procs=. then rmw_procs=0;
    if o_procs=. then o_procs=0;

   tot_exp=sum(fcc_cost,pp_cost,ar_cost,ie_cost,tax_cost);
   tot_cost=sum(int_cost,tot_exp,last_upb);
   tot_proc=sum(ns_procs,ce_procs,rmw_procs,o_procs);

  end;

run;

%mend;

%LPPUB(2000Q1);
%LPPUB(2000Q2);
%LPPUB(2000Q3);
```

```
%LPPUB(2000Q4);

%LPPUB(2001Q1);
%LPPUB(2001Q2);
%LPPUB(2001Q3);
%LPPUB(2001Q4);

%LPPUB(2002Q1);
%LPPUB(2002Q2);
%LPPUB(2002Q3);
%LPPUB(2002Q4);

%LPPUB(2003Q1);
%LPPUB(2003Q2);
%LPPUB(2003Q3);
%LPPUB(2003Q4);

%LPPUB(2004Q1);
%LPPUB(2004Q2);
%LPPUB(2004Q3);
%LPPUB(2004Q4);

%LPPUB(2005Q1);
%LPPUB(2005Q2);
%LPPUB(2005Q3);
%LPPUB(2005Q4);

%LPPUB(2006Q1);
%LPPUB(2006Q2);
%LPPUB(2006Q3);
%LPPUB(2006Q4);

%LPPUB(2007Q1);
%LPPUB(2007Q2);
%LPPUB(2007Q3);
%LPPUB(2007Q4);

%LPPUB(2008Q1);
%LPPUB(2008Q2);
%LPPUB(2008Q3);
%LPPUB(2008Q4);

%LPPUB(2009Q1);
%LPPUB(2009Q2);
%LPPUB(2009Q3);
%LPPUB(2009Q4);

%LPPUB(2010Q1);
%LPPUB(2010Q2);
%LPPUB(2010Q3);
%LPPUB(2010Q4);

%LPPUB(2011Q1);
%LPPUB(2011Q2);
```

```
%LPPUB(2011Q3);
%LPPUB(2011Q4);

%LPPUB(2012Q1);
%LPPUB(2012Q2);
%LPPUB(2012Q3);
%LPPUB(2012Q4);

%LPPUB(2013Q1);
%LPPUB(2013Q2);
%LPPUB(2013Q3);
%LPPUB(2013Q4);

%LPPUB(2014Q1);
%LPPUB(2014Q2);

data myfolder.combined_data;
   set temp.comb_2000q1 temp.comb_2000q2 temp.comb_2000q3 temp.comb_2000q4
       temp.comb_2001q1 temp.comb_2001q2 temp.comb_2001q3 temp.comb_2001q4
       temp.comb_2002q1 temp.comb_2002q2 temp.comb_2002q3 temp.comb_2002q4
       temp.comb_2003q1 temp.comb_2003q2 temp.comb_2003q3 temp.comb_2003q4
       temp.comb_2004q1 temp.comb_2004q2 temp.comb_2004q3 temp.comb_2004q4
       temp.comb_2005q1 temp.comb_2005q2 temp.comb_2005q3 temp.comb_2005q4
       temp.comb_2006q1 temp.comb_2006q2 temp.comb_2006q3 temp.comb_2006q4
       temp.comb_2007q1 temp.comb_2007q2 temp.comb_2007q3 temp.comb_2007q4
       temp.comb_2008q1 temp.comb_2008q2 temp.comb_2008q3 temp.comb_2008q4
       temp.comb_2009q1 temp.comb_2009q2 temp.comb_2009q3 temp.comb_2009q4
       temp.comb_2010q1 temp.comb_2010q2 temp.comb_2010q3 temp.comb_2010q4
       temp.comb_2011q1 temp.comb_2011q2 temp.comb_2011q3 temp.comb_2011q4
       temp.comb_2012q1 temp.comb_2012q2 temp.comb_2012q3 temp.comb_2012q4
       temp.comb_2013q1 temp.comb_2013q2 temp.comb_2013q3 temp.comb_2013q4
       temp.comb_2014q1 temp.comb_2014q2;
run;

  proc sort data = myfolder.combined_data; by loan_id; run;

***********************************;
********start of analysis********;
***********************************;

**************************************;
*** Attribute Formats Used in Tables **;
**************************************;

proc format;
   value cltvfmt

    0-60    = '(0-60]'
    60<-70  = '(60-70]'
    70<-75  = '(70-75]'
    75<-80  = '(75-80]'
    80<-85  = '(80-85]'
    85<-90  = '(85-90]'
    90<-95  = '(90-95]'
    95<-97  = '(95-97]'
    97<-998 = '(97+]'
    999      = 'All'
    ;

   value cscorefmt

    780-high = '[780+)'
    740-<780 = '[740-780)'
    700-<740 = '[700-740)'
    660-<700 = '[660-700)'
    620-<660 = '[620-660)'
    low-<620 = '[0-620)'
    .           = 'missing'

    ;

   value dtifmt
      low-<20 = '[0-20)'
      20-<30 = '[20-30)'
      30-<40 = '[30-40)'
      40-<45 = '[40-45)'
      45-high  = '[45+]'
      other          = 'Missing'
;

   value borrfmt
      1 = '1'
      2 = '2'
      3 - high = '3+'
      other    = 'Missing'
;

value  upbfmt
      0 - 85000 = '$0.00 - $85,000'
      85000  <- 110000 = '$85,000 - $110,000'
      110000 <- 125000 = '$110,000 - $125,000'
      125000 <- 150000 = '$125,000 - $150,000'
      150000 <- 175000 = '$150,000 - $175,000'
      175000 <- 200000 = '$175,000 - $200,000'
      200000 <- 417000 = '$200,000 - $417,000'
      417000 <- high   = '$417,000+'
      other = 'Missing';

run;

*****************************************************************************************************
**********************************;
********************************** Historical Loan Level Data - Format Dataset Created in Data 101
*******************************;
*****************************************************************************************************
**********************************;

data hist;
```

```
	set myfolder.combined_data;

	o_wght = orig_amt / 1000000;

	** Set Missing OCLTVs to OLTV **;
	if OCLTV =. then OCLTV = OLTV;

	** Identify Risk Layers **;
	if PURPOSE = "C" then cor_flg = 1; else cor_flg = 0;
	if OCC_STAT = "I" then invest_flg = 1; else invest_flg = 0;
	if DTI=. then dti_flg = 1; else dti_flg = 0;
	if DTI > 45 then dti_flg = 1; else dti_flg = 0;
	if NUM_BO = 1 then oneborr_flg = 1; else oneborr_flg = 0;

	risklayers=sum(cor_flg,invest_flg,dti_flg,oneborr_flg);

	** Comp Bucketing/Key **;
	length key $ 20 ;
	key = cats(put(CSCORE_MN,cscorefmt.),"_",put(OCLTV,cltvfmt.),"_",put(risklayers,$1.));

	*active upb;
	if LAST_STAT not in ("F","S","P","R") then actv_upb = LAST_UPB; else actv_upb = 0;
	pool_fctr = actv_upb/orig_amt;

	** reo completion flag **;
	*building a flag to designate loans we consider part of our defaulted loan population;
	if last_stat in ("F","S") and disp_dte ne . then complt_flg = 1; else complt_flg= 0;
	*default upb is the final reported upb for loans in our defaulted loan population;
	if complt_flg = 1 then default_upb = last_upb; else default_upb = 0;

	** performance rates **;
	*calculating the net credit event rate, the portion of our originations that defaults, and the portion of originations
that we lose due to default;
	nce_upb = fce_upb;
	if LAST_STAT = "R" or intck("month",frst_pay,fce_dte) > 120 then nce_upb = 0;
	nce_rt = nce_upb/orig_amt;
	if nce_rt =. then nce_rt =0;

	def_rt = default_upb/orig_amt;

	if complt_flg = 1 then do;
		nloss_rt = net_loss/orig_amt;
		severity = net_loss/default_upb;
	end;

	** Convert Number of Borrowers to Numeric**;
	borr=input(NUM_BO,best.);

	** Create YYYYQQ merge keys for MTM LTV **;

	length orig_mrg $ 6 last_mrg $ 6;

	orig_mrg=cats(put(year(ORIG_DTE),$4.),"_",put(qtr(ORIG_DTE),$1.));
	last_mrg=cats(put(year(LAST_DTE),$4.),"_",put(qtr(LAST_DTE),$1.));
```

```
run;

data sasdata;
       set hist (where = (nce_upb >0 and OLTV>80 and
                                      OLTV<=97 and
                                      OCLTV<=97));
run;


******************************************************************************************************
*******************************;
******************************* Performance Tables - Historical Default, Severity, & Loss Rates
*******************************;
******************************************************************************************************
*******************************;

** Performance Table Macro **;

%macro tables (class,class2,format,title);
proc tabulate data = hist (where = (year(ORIG_DTE)<2013)) missing;
class &class ORIG_DTE;
format &class2 &format. ORIG_DTE year.;
var def_rt/weight=o_wght;
table (&class='' all), (ORIG_DTE='' all)*def_rt=''*mean=''*f=percent10.2;
title "Historical Default Performance by &title.";
run;

proc tabulate data = hist (where = (year(ORIG_DTE)<2013)) missing;
class &class ORIG_DTE;
format &class2 &format. ORIG_DTE year.;
var severity/weight=default_upb;
table (&class='' all), (ORIG_DTE='' all)*severity=''*mean=''*f=percent10.2;
title "Historical Severity by &title.";
run;

proc tabulate data = hist (where = (year(ORIG_DTE)<2013)) missing;
class &class ORIG_DTE;
format &class2 &format. ORIG_DTE year.;
var nloss_rt/weight=o_wght;
table (&class='' all), (ORIG_DTE='' all)*nloss_rt=''*mean=''*f=percent10.2;
title "Historical Loss Performance by &title.";
run;
%mend;

******************************************************************************************************
**********************************;
************************************************* Performance Comp using CAS Deal Profile
********************************************;
******************************************************************************************************
**********************************;

** CAS Deal Tape Header **;
```

```
%let cas_head =
POOL_ID
LOAN_ID :$10.
ACT_DTE :$06.
CHANNEL :$01.
SELLER  :$50.
SERVICER :$50.
MSTR_SERV :$10.
ORIG_RTE
CURR_RTE
ORIG_UPB
ISSUE_UPB
CURR_UPB
TERM
ORIG_DTE :$06.
FIRST_DTE :$06.
LOAN_AGE
REM_MTHS
AREM_MTHS
MATR_DTE :$06.
OLTV
OCLTV
NUM_BO
DTI
CSCOREB
CSCOREC
FTHB :$01.
PURPOSE :$21.
PROP_TYP :$06.
NUM_UNITS
OCC_STAT :$09.
STATE :$02.
MSA :$05.
ZIP_3 :$03.
MI_PCT
PRODUCT :$03.
PRE_PENALTY :$01.
IO_FLAG :$01.
IO_DTE :$06.
IO_MTHS
DLQ_STAT :$02.
PMT_HIST :$48.
MOD_FLAG :$01.
MI_CANCEL :$02.
ZB_CODE :$03.
ZB_DATE :$06.
LAST_UPB
RP_DATE :$06.
SCHED_PRINC
TOTAL_PRINC
UNSCH_PRINC
;
**** To Access CAS 2015-C02 Group 1 Deal Tape ****;
**** Wells Fargo Corporate Trust Link - https://www.ctslink.com/ ****;
```

```
**** Register with site, Select "Series" from search drop down, Search "CAS" ****;
**** Select CAS 2015-C02, select "Deal Documents Tab," Download and Save CAS 2015-C02 At Issuance File to
local directory";
**** Group 1 Deal Tape: FNMA_2015C02_PLDT1.csv ****;

** 1. Import CAS 2015-C02 Group 1 & 2 Loan Tape, Create Merge Key **;
data CAS15C02G1;
	infile "./FNMA_2015C02_PLDT1.csv" delimiter="|" MISSOVER DSD lrecl=32767;
	input &cas_head;

	** Identify Risk Layers **;
	if PURPOSE = "CASH-OUT REFINANCE" then cor_flg = 1; else cor_flg = 0;
	if OCC_STAT = "Investor" then invest_flg = 1; else invest_flg = 0;
	if DTI=. then dti_flg = 1; else dti_flg = 0;
	if DTI > 45 then dti_flg = 1; else dti_flg = 0;
	if NUM_BO = 1 then oneborr_flg = 1; else oneborr_flg = 0;

	risklayers=sum(cor_flg,invest_flg,dti_flg,oneborr_flg);

	** Min FICO **;
	if CSCOREC = . then MIN_FICO = CSCOREB; else MIN_FICO = min(CSCOREC,CSCOREB);

	** Comp Bucketing/Key **;
	length key $ 20 ;
	key = cats(put(MIN_FICO,cscorefmt.),"_",put(OCLTV,cltvfmt.),"_",put(risklayers,$1.));
run;

data CAS15C02G2;
	infile "./FNMA_2015C02_PLDT2.csv" delimiter="|" MISSOVER DSD lrecl=32767;
	input &cas_head;

	** Identify Risk Layers **;
	if PURPOSE = "CASH-OUT REFINANCE" then cor_flg = 1; else cor_flg = 0;
	if OCC_STAT = "Investor" then invest_flg = 1; else invest_flg = 0;
	if DTI=. then dti_flg = 1; else dti_flg = 0;
	if DTI > 45 then dti_flg = 1; else dti_flg = 0;
	if NUM_BO = 1 then oneborr_flg = 1; else oneborr_flg = 0;

	risklayers=sum(cor_flg,invest_flg,dti_flg,oneborr_flg);

	** Min FICO **;
	if CSCOREC = . then MIN_FICO = CSCOREB; else MIN_FICO = min(CSCOREC,CSCOREB);

	** Comp Bucketing/Key **;
	length key $ 20 ;
	key = cats(put(MIN_FICO,cscorefmt.),"_",put(OCLTV,cltvfmt.),"_",put(risklayers,$1.));
run;

** 2a.Proc Tabulate to Produce Historical Net Credit Event Rate **;

	proc tabulate data = hist (where=(OLTV>60 and
	                                   OLTV<=80 and
	                                   OCLTV<=97)) out= hist_prf_grp1;
	class ORIG_DTE key;
```

```
        format ORIG_DTE year.;
        var nce_rt/weight=ORIG_AMT;
        table (key='' all)*ORIG_DTE='', (nce_rt)*mean=''*f=best32.;
        run;

        proc tabulate data = hist (where=(OLTV>80 and
                                  OLTV<=97 and
                                  OCLTV<=97)) out= hist_prf_grp2;
        class ORIG_DTE key;
        format ORIG_DTE year.;
        var nce_rt/weight=ORIG_AMT;
        table (key='' all)*ORIG_DTE='', (nce_rt)*mean=''*f=best32.;
        run;

** 2b. Proc Tabulate to Produce CAS UPB Distribution **;

        proc tabulate data = CAS15C02G1 out=cas_distr_grp1;
        class key;
        var ISSUE_UPB;
        table key='', ISSUE_UPB=''*colpctsum=''*f=best32.;
        run;

        proc tabulate data = CAS15C02G2 out=cas_distr_grp2;
        class key;
        var ISSUE_UPB;
        table key='', ISSUE_UPB=''*colpctsum=''*f=best32.;
        run;

** 2c. Sort Tabular Output for Merge **;

        proc sort data = hist_prf_grp1;
        by key;
        run;

        proc sort data = cas_distr_grp1;
        by key;
        run;

        proc sort data = hist_prf_grp2;
        by key;
        run;

        proc sort data = cas_distr_grp2;
        by key;
        run;

** 3. Merge Historical Rates output to CAS UPB Distribution **;
        data comp_grp1;
        merge hist_prf_grp1  (in = a drop = _TYPE_ _PAGE_ _TABLE_)
            cas_distr_grp1 (in = b drop = _TYPE_ _PAGE_ _TABLE_);
            if a and b;
            by key;

            ** Comped Net Credit Event Rate **;
            comp_nce = nce_rt_Mean*ISSUE_UPB_PctSum_0/100;
        run;

        data comp_grp2;
        merge hist_prf_grp2  (in = a drop = _TYPE_ _PAGE_ _TABLE_)
            cas_distr_grp2 (in = b drop = _TYPE_ _PAGE_ _TABLE_);
            if a and b;
            by key;

            ** Comped Net Credit Event Rate **;
            comp_nce = nce_rt_Mean*ISSUE_UPB_PctSum_0/100;
        run;

** 4. Calculate Fixed Severity Loss Outcomes  **;
proc tabulate data = hist (where=(OLTV>60 and OLTV<=80 and OCLTV<=97)) out = hist_tab_grp1;
        class ORIG_DTE;
        format ORIG_DTE year.;
        var actv_upb;
        var pool_fctr nce_rt / weight = orig_amt;
        table ORIG_DTE='', actv_upb=''*sum='Remaining UPB'
                                  pool_fctr=''*mean='Pool Factor'*f=percent10.5
                                  nce_rt=''*mean='Net CE Rate'*f=percent10.5;
        title "Historical Default & Loss Comped to CAS 2015-C02 Group 1";
run;

proc tabulate data = hist (where=(OLTV>80 and OLTV<=97 and OCLTV<=97)) out = hist_tab_grp2;
        class ORIG_DTE;
        format ORIG_DTE year.;
        var actv_upb;
        var pool_fctr nce_rt / weight = orig_amt;
        table ORIG_DTE='', actv_upb=''*sum='Remaining UPB'
                                  pool_fctr=''*mean='Pool Factor'*f=percent10.5
                                  nce_rt=''*mean='Net CE Rate'*f=percent10.5;
        title "Historical Default & Loss Comped to CAS 2015-C02 Group 2";
run;

data hist_comp_tab_grp1 (drop = CDealStrctLss1 CDealStrctLss2 CDealStrctLss3 _TYPE_ _PAGE_ _TABLE_);
        set hist_tab_grp1;

        if nce_rt_Mean>0 then CDealStrctLss1 = min(0.01,nce_rt_Mean)*0.1; else CDealStrctLss1 = 0;
        if (nce_rt_Mean-0.01) >0 then CDealStrctLss2 = min(0.01,(nce_rt_Mean-0.01))*0.2; else CDealStrctLss2 = 0;
        if (nce_rt_Mean-0.02) >0 then CDealStrctLss3 = (nce_rt_Mean-0.02)*0.4; else CDealStrctLss3 = 0;

        CDealStrctLss = CDealStrctLss1 + CDealStrctLss2 + CDealStrctLss3;
run;

data hist_comp_tab_grp2 (drop = CDealStrctLss1 CDealStrctLss2 CDealStrctLss3 _TYPE_ _PAGE_ _TABLE_);
        set hist_tab_grp2;

        if nce_rt_Mean>0 then CDealStrctLss1 = min(0.01,nce_rt_Mean)*0.1; else CDealStrctLss1 = 0;
        if (nce_rt_Mean-0.01) >0 then CDealStrctLss2 = min(0.02,(nce_rt_Mean-0.01))*0.2; else CDealStrctLss2 = 0;
        if (nce_rt_Mean-0.03) >0 then CDealStrctLss3 = (nce_rt_Mean-0.03)*0.25; else CDealStrctLss3 = 0;

        CDealStrctLss = CDealStrctLss1 + CDealStrctLss2 + CDealStrctLss3;
```

```
run;

proc tabulate data = comp_grp1 out = comp_tab_grp1;
	class ORIG_DTE;
	format ORIG_DTE year.;
	var comp_nce;
	table ORIG_DTE='', (comp_nce)*sum=''*f=percent10.5;
	title "Historical Default & Loss Comped to CAS 2015-C01";
run;

data deal_comp_tab_grp1 (drop = CDealStrctLss1 CDealStrctLss2 CDealStrctLss3 _TYPE_ _PAGE_ _TABLE_);
	set comp_tab_grp1;
	if comp_nce_Sum>0 then CDealStrctLss1 = min(0.01,comp_nce_Sum)*0.1; else CDealStrctLss1 = 0;
	if (comp_nce_Sum-0.01) >0 then CDealStrctLss2 = min(0.01,(comp_nce_Sum-0.01))*0.2; else CDealStrctLss2 =
0;
	if (comp_nce_Sum-0.02) >0 then CDealStrctLss3 = (comp_nce_Sum-0.02)*0.4; else CDealStrctLss3 = 0;

	CDealStrctLss = CDealStrctLss1 + CDealStrctLss2 + CDealStrctLss3;
run;

proc tabulate data = comp_grp2 out = comp_tab_grp2;
	class ORIG_DTE;
	format ORIG_DTE year.;
	var comp_nce;
	table ORIG_DTE='', (comp_nce)*sum=''*f=percent10.5;
	title "Historical Default & Loss Comped to CAS 2015-C02";
run;

data deal_comp_tab_grp2 (drop = CDealStrctLss1 CDealStrctLss2 CDealStrctLss3 _TYPE_ _PAGE_ _TABLE_);
	set comp_tab_grp2;
	if comp_nce_Sum>0 then CDealStrctLss1 = min(0.01,comp_nce_Sum)*0.1; else CDealStrctLss1 = 0;
	if (comp_nce_Sum-0.01) >0 then CDealStrctLss2 = min(0.02,(comp_nce_Sum-0.01))*0.2; else CDealStrctLss2 =
0;
	if (comp_nce_Sum-0.03) >0 then CDealStrctLss3 = (comp_nce_Sum-0.03)*0.25; else CDealStrctLss3 = 0;

	CDealStrctLss = CDealStrctLss1 + CDealStrctLss2 + CDealStrctLss3;
run;

proc print data = hist_comp_tab_grp1;
run;

proc print data = deal_comp_tab_grp1;
run;

proc print data = hist_comp_tab_grp2;
run;

proc print data = deal_comp_tab_grp2;
run;

*******************************************************************************************************
*******************************;
************************************ Mark-to-Market LTV Using FHFA 3-Digit Zip Code HPI
************************************;
```

```
*****************************************************************************************************
*****************************;


**** Download HPI from FHFA website and save file as CSV ****;
**** http://www.fhfa.gov/DataTools/Downloads/Documents/HPI/HPI_AT_3zip.xlsx ****;

***************************;
** 1. Import FHFA Zip-3 HPI **;
***************************;

%let head =

ZIP_3 :$3.
year 4.
quarter 1.
index 6.2
type :$20.
;

  data hpi (where =(year>1998) drop = type);
    infile "./HPI_AT_3zip_2015Q1.csv" delimiter="," MISSOVER DSD lrecl=32767 firstobs=6;
      input &head;
      length orig_mrg $6. last_mrg $6.;
      orig_mrg=cats(put(year,$4.),"_",put(quarter,$1.));
      last_mrg=orig_mrg;

  run;

**********************************************************;
** Double Merge on HPI to get orig and last index values **;
**********************************************************;

proc sort data = hist;
      by ZIP_3 orig_mrg;
run;
proc sort data = hpi;
      by ZIP_3 orig_mrg;
run;

** 2. First Merge for Origination HPI value **;

data merge1;
  merge hist (in=a where=(year(ORIG_DTE)=2007))
       hpi (in = b keep = ZIP_3 orig_mrg index rename=(index=base_hpi));
       if a;
       by ZIP_3 orig_mrg;
run;

proc sort data = merge1;
      by ZIP_3 last_mrg;
run;

proc sort data = hpi;
```

```
       by ZIP_3 last_mrg;
run;

** 3. Second Merge for Last HPI value - Calculate MTM LTV **;
data mltv;
   merge merge1 (in=a)
        hpi (in=b keep = ZIP_3 last_mrg index rename=(index=last_hpi));
        if a;
        by ZIP_3 last_mrg;

        ** MLTV Calculation **;

        hpi_factor = last_hpi/base_hpi;
        LAST_VAL = hpi_factor*ORIG_VAL;
        mltv       = last_upb/LAST_VAL;
run;

************************************************************************************************************************************;
********************************************************************* Output to Excel
*********************************************************************;
************************************************************************************************************************************;
ods tagsets.excelxp file = "./Data 102 Performance Tables.xls"
                 style=Seaside
                 options ( FitToPage       = 'yes'
                           Pages_FitWidth  = '1'
                           Pages_FitHeight = '1'
                           Autofit_Height  = 'Yes'
                           );

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='Occupancy');
%tables(OCC_STAT,,,Occupancy);

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='Purpose');
%tables(Purpose,,,Purpose);

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='Borrowers');
%tables(borr,borr,borrfmt.,Number of Borrowers);

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='MinFICO');
%tables(cscore_mn,cscore_mn,cscorefmt.,Min FICO);

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='Orig UPB');
%tables(orig_amt,orig_amt,upbfmt.,Loan Amount);

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='CLTV');
%tables(OCLTV,OCLTV,cltvfmt.,CLTV);

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='DTI');
%tables(DTI,DTI,dtifmt.,DTI);

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='Risk Layers');
%tables(risklayers,,,Risk Layers);
```

```
ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='LTVxFICO');
proc tabulate data = hist (where = (year(ORIG_DTE)=2006)) missing;
class OLTV cscore_mn;
format OLTV cltvfmt. cscore_mn cscorefmt.;
var def_rt/weight=o_wght;
table (cscore_mn='' all), (OLTV='' all)*def_rt=''*mean=''*f=percent10.2;
title "Historical Default Performance by LTV & FICO for 2006 Vintage";
run;

proc tabulate data = hist (where = (year(ORIG_DTE)=2006)) missing;
class OLTV cscore_mn;
format OLTV cltvfmt. cscore_mn cscorefmt.;
var severity/weight=default_upb;
table (cscore_mn='' all), (OLTV='' all)*severity=''*mean=''*f=percent10.2;
title "Historical Severity by LTV & FICO for 2006 Vintage";
run;

proc tabulate data = hist (where = (year(ORIG_DTE)=2006)) missing;
class OLTV cscore_mn;
format OLTV cltvfmt. cscore_mn cscorefmt.;
var nloss_rt/weight=o_wght;
table (cscore_mn='' all), (OLTV='' all)*nloss_rt=''*mean=''*f=percent10.2;
title "Historical Loss Performance by LTV & FICO for 2006 Vintage";
run;

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='210 Cohorts');
proc tabulate data = hist (where = (year(ORIG_DTE)=2007)) missing;
class OLTV cscore_mn risklayers;
format OLTV cltvfmt. cscore_mn cscorefmt.;
var orig_amt;
table (cscore_mn='FICOBkt'*risklayers='RskBkt') all, pctsum=''*orig_amt=''*(OLTV='' all);
title "2007 Origination Vintage UPB Distribution";
run;

proc tabulate data = hist (where = (year(ORIG_DTE)=2007)) missing;
class OLTV cscore_mn risklayers;
format OLTV cltvfmt. cscore_mn cscorefmt.;
var def_rt/weight=o_wght;
table (cscore_mn='FicoBkt' all)*(risklayers='RskFctrs' all),(OLTV='' all)*def_rt=''*mean=''*f=percent10.2;
title "2007 Origination Vintage Default Rate";
run;

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='CAS2015-C02 Group I Comp');
proc print data = hist_comp_tab_grp1;
run;

proc print data = deal_comp_tab_grp1;
run;

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='CAS2015-C02 Group II Comp');
proc print data = hist_comp_tab_grp2;
run;
```

```
proc print data = deal_comp_tab_grp2;
run;

ods tagsets.excelxp options(sheet_interval = 'none' sheet_name='MTMLTV');
proc tabulate data = mltv;
	class LAST_STAT;
	var mltv OLTV / weight = orig_amt;
	table (LAST_STAT='' all), OLTV*mean=''*f=percent10.5 mltv='MLTV'*mean=''*f=percent10.5;
run;

ods tagsets.excelxp close;
```