

set printback=on.

\*\*\*\*\*

CFSR 3: OBSERVED PERFORMANCE  
- MALTREATMENT IN FOSTER CARE.

\*\*\*\*\*

\* Before running this syntax, ensure you have:

\* 1. Unzipped the CFSR3IndicatorsSyntax.zip file available from the Children's Bureau. Unzipping this file will

\* create the folders and provide some SPSS files needed to run this syntax.

\* 2. Placed the following file(s) in the 'CFSR3\Submissions NCANDS\' folder:

\* AllStateYYChildFiles.sav

\* AllStateYYChildFiles.sav is a merged file containing all states' NCANDS child file submissions for a given FY,

\* where YY represents a FY. The submissions folder should contain, at a minimum, the

\* AllStateYYChildFiles.sav file for the FY associated with the cohort of children for whom performance will be

\* examined. For example, if you are interested in examining maltreatment in foster for children served in foster

\* care during 13A13B, you will need a file called AllState13ChildFiles.sav which contains NCANDS

\* submissions associated with FY13.

\* 3. Run CFSR 3 - 1 Create AFCARS source data.sps

\* 4. Run CFSR 3 - 2 Create AFCARS DQ files.sps

\* Manual input is needed in sections that start and end with the following:

\*\*\*\*\* START USER INPUT \*\*\*\*\*

\*\*\*\*\* END USER INPUT \*\*\*\*\*

\*\*\*\*\*

\* GET SOURCE FILE

\*\*\*\*\*

\* Set working directory.

cd 'C:\CFSR3\Analysis - FR 2015 Apr Replicate'.

show directory.

\* Open AFCARS source data.

get file 'CFSR 3 AFCARS source data.sav'.

dataset name SourceData window=front.

\* Prevent accidentally overwriting source data.

dataset copy Indicator.

dataset activate Indicator.  
dataset close SourceData.

\* Delete unused variables (Note: recnumbr is needed to support matching with NCANDS file).

delete variables amiakn asian blkafam hawaiiipi white untodetm hisorgin clindis mr vishear phydis dsmlll othermed everadpt ageadopt manrem phyabuse sexabuse neglect  
aaparent daparent aachild dachild childis chbehprb prtsdied prtsjail nocope abandmnt relinqsh housing placeout casegoal ctkfamst ctk1yr ctk2yr fosfamst fctk1yr fcctk2yr  
rf1amakn rf1asian rf1blkaa rf1nhopi rf1white rf1utod hofcctk1 rf2amakn rf2asian rf2blkaa rf2nhopi rf2white rf2utod hofcctk2 ivefc iveaa ivaafdc ivdchsup xixmedcd ssioter  
noa fcmntpay DtReview  
DtLatRemTrans DtTPRMom DtTPRDad DtDischTrans.

\*\*\*\*\*

\* SPECIFY THE 12-MONTH COHORT WHOSE OUTCOME WILL BE ASSESSED

\*\*\*\*\*

\* Identify the 12-month cohort of interest (e.g., children in care during 13A13B) by entering "AB" or "BA" for

\* PeriodType and the year the 12-month period ends (for YYYY and YYstr).

\* AB period (A file + B file) spans Oct 1 - Sept 30 of the following year.

\* BA period (B file + A file) spans Apr 1 - Mar 31 of the following year.

\* Examples:

\* ( \* ) represents a FY

\* Children in care during PeriodType YYYY YYstr

* 09B10A	BA	2010	10
* 10A10B *	AB	2010	10
* 10B11A	BA	2011	11
* 11A11B *	AB	2011	11
* 11B12A	BA	2012	12
* 12A12B *	AB	2012	12
* 12B13A	BA	2013	13
* 13A13B *	AB	2013	13
* 13B14A	BA	2014	14

\* etc.

\*\*\*\*\* START USER INPUT \*\*\*\*\*

define PeriodType () "AB"

!enddefine.

define YYYY () 2013.

!enddefine.

define YYstr () "13"

!enddefine.

\*\*\*\*\* END USER INPUT \*\*\*\*\*

\* Create variable that indicates the user-specified 12-month cohort (e.g., "AB13" represents "13A13B").  
string TwelveMoCohort (A4).  
compute TwelveMoCohort = concat(PeriodType,YYstr).  
execute.

\*\*\*\*\*

#### CREATE DATE PARAMETERS

\*\*\*\*\*

\* Start date of a 12-month period (10/1/YYYY or 4/1/YYYY).  
\* End date of a 12-month period (3/31/YYYY or 9/30/YYYY).

\* Maltreatment in foster care involves following children who were served during the 12-month period specified  
\* earlier. Served includes children who entered care during the period, or were in care on the first day of the  
\* period. Identify the start date (DtPeriodBeg) and end date (DtPeriodEnd) that defines this 12-month period.

do if PeriodType = "AB".  
compute DtPeriodBeg=date.mdy(10,01,YYYY-1).  
compute DtPeriodEnd=date.mdy(09,30,YYYY).  
end if.  
do if PeriodType = "BA".  
compute DtPeriodBeg=date.mdy(04,01,YYYY-1).  
compute DtPeriodEnd=date.mdy(03,31,YYYY).  
end if.  
execute.  
formats DtPeriodBeg DtPeriodEnd (adate10).

#### variable labels

TwelveMoCohort '12-month period children were served and for whom performance is being assessed'  
DtPeriodBeg 'Start date of the 12-month period specified'  
DtPeriodEnd 'End date of the 12-month period specified'.

\*\*\*\*\*

#### \* REMOVE STATES THAT EXCEED DQ LIMITS FOR THE AFCARS DQ CHECKS

\*\*\*\*\*

\* The AFCARS cross-file checks (Dropped records and AFCARS IDs don't match from one period to the next)  
\* apply only to the first 6-month period (DtReportBeg = DtPeriodBeg). The AFCARS within-file checks apply to  
\* both periods (DtReportBeg ge DtPeriodBeg AND DtReportBeg le DtPeriodEnd). The NCANDS DQ checks  
\* will be applied later.

\* Open AFCARS data quality results.  
get file 'DQ AFCARS\Merged\Merged AFCARS DQ files.sav'.  
dataset name DQResults window=front.

\* Prevent accidentally overwriting file.

dataset copy DQIndicator.

dataset activate DQIndicator.

dataset close DQResults.

\* Merge in variables from the Indicator file, then keep only DtPeriodBeg and DtPeriodEnd.

match files

/FILE=\*

/FILE='Indicator'

/KEEP state to DQFailedCheck DtPeriodBeg DtPeriodEnd.

execute.

\* Select the AFCARS checks for this indicator and DQ results for the 6-month period(s).

\* ... Cross file checks (apply only to the first 6-month period).

if MALFC = 1 and DQFileXW = "Cross file" AND (DtReportBeg eq DtPeriodBeg) DQChecksApply = 1.

\* ... Within file checks (apply to both 6-month periods).

if MALFC = 1 and DQFileXW = "Within file" AND (DtReportBeg ge DtPeriodBeg) and (DtReportEnd le DtPeriodEnd) DQChecksApply = 1.

execute.

\* Review the DQ checks that will be applied for this indicator.

temporary.

select if DQChecksApply = 1.

ctables

/VLABELS VARIABLES=DQCheckShort DISPLAY=NONE /VLABELS VARIABLES=SixMoPeriod DQChecksApply

DISPLAY=LABEL

/TABLE DQCheckShort [C] BY SixMoPeriod [C] > DQChecksApply [S][MAXIMUM]

/SLABELS VISIBLE=NO

/CATEGORIES VARIABLES=DQCheckShort SixMoPeriod ORDER=A KEY=VALUE EMPTY=EXCLUDE.

\* For each state, for the checks that apply, sum the number of checks it failed.

sort cases by state.

select if DQChecksApply = 1.

execute.

aggregate

/OUTFILE=\* MODE=ADDVARIABLES

/BREAK=state

/DQFailedCheck\_sum=SUM(DQFailedCheck).

\* If DQFailedCheck\_sum > 0, then state failed at least one of the checks for at least one of the 6-month periods.

\* List those states, the checks they failed, the % of problem cases, and the periods in which the checks failed.

temporary.

select if DQFailedCheck\_sum > 0 and DQFailedCheck = 1.

ctables

/VLABELS VARIABLES=stateabb DQCheckShort DQCheckLong DISPLAY=NONE /VLABELS VARIABLES=DQLimit SixMoPeriod

DQResult DISPLAY=BOTH

```
/TABLE stateabb [C] > DQCheckShort [C] > DQCheckLong [C] BY DQLimit [S][MAXIMUM] + SixMoPeriod [C] >
  DQResult [S][MAXIMUM]
/SLABELS VISIBLE=NO
/CATEGORIES VARIABLES=stateabb DQCheckShort DQCheckLong SixMoPeriod ORDER=A KEY=VALUE EMPTY=EXCLUDE
/TITLES
  TITLE='For Maltreatment in foster care, performance was not calculated for the following states ' ' due to exceeding the DQ limits for the following checks and 6-month periods'.
```

\* Merge states' DQFailedCheck\_sum value into Indicator file.

\* Any state where DQFailedCheck\_sum > 0 will be dropped from the dataset and excluded from analyses.

\* ... Select only one record for each state (avoids duplicate key error).

sort cases BY state(A).

match files

/FILE=\*

/BY state

/LAST=PrimaryLast.

VARIABLE LABELS PrimaryLast 'Indicator of each last matching case as Primary'.

VALUE LABELS PrimaryLast 0 'Duplicate Case' 1 'Primary Case'.

VARIABLE LEVEL PrimaryLast (ORDINAL).

execute.

select if PrimaryLast = 1.

execute.

\* ... Merge in all variables from DQIndicator file, then keep only DQFailedCheck\_sum.

dataset activate Indicator.

sort cases by state.

match files

/FILE=\*

/TABLE='DQIndicator'

/BY state

/KEEP state to DtPeriodEnd DQFailedCheck\_sum.

execute.

\* Verify once more the states about to be excluded.

temporary.

select if DQFailedCheck\_Sum > 0.

crosstabs

/TABLES=stateabb BY DQFailedCheck\_sum

/FORMAT=AVALUE TABLES

/CELLS=COUNT

/COUNT ROUND CELL.

dataset close DQIndicator.

\* Select only states that met the DQ checks.

```
select if DQFailedCheck_sum = 0.  
execute.  
delete variables DQFailedCheck_sum.
```

```
* Count the number of states remaining.  
compute numstates = 1.  
if (state eq lag(state))numstates = 0.  
frequencies numstates.
```

```
*****  
* SELECT APPLICABLE RECORDS  
*****
```

```
* Select only 6-month records that were reported during the 12-month period (between DtPeriodBeg and  
* DtPeriodEnd).
```

```
*****
```

```
* Flag the records to keep.  
if (DtReportBeg ge DtPeriodBeg) and (DtReportEnd le DtPeriodEnd) ReportedDuringPeriod = 1.  
execute.
```

```
* Verify the correct six 6-month periods have been selected.
```

```
crosstabs  
/TABLES=DtReportBeg BY ReportedDuringPeriod  
/FORMAT=AVALUE TABLES  
/CELLS=COUNT  
/COUNT ROUND CELL.
```

```
* Select the records to keep.  
select if ReportedDuringPeriod=1.  
execute.
```

```
delete variables ReportedDuringPeriod.
```

```
* Select only the most recent 6-month record reported for each child, for each episode.
```

```
*****
```

```
* This will create an episode-level file. Episodes are distinguished by the date of latest removal from home.  
* A child has a record in each 6-month submission until he discharged, dropped, or was still in care as of  
* the last reporting period we have in the file. We only want one record per child, per episode, and the record we  
* pick is the last one reported for that episode. This record will contain the most recent data available that  
* describes the child's episode (e.g., LOS, age, etc.).
```

```
* Flag the records to keep.  
sort cases BY ChildID (A) DtLatRem (A).
```

match files  
/FILE=\*  
/BY ChildID DtLatRem  
/LAST=Flag\_LastRpt4Ep.  
variable labels Flag\_LastRpt4Ep 'last 6-month report we received for this episode (based on DtLatRem)'.  
value labels Flag\_LastRpt4Ep 0 'Duplicate Case' 1 'Primary Case'.  
variable level Flag\_LastRpt4Ep (ORDINAL).  
frequencies variables=Flag\_LastRpt4Ep.  
execute.

\* Select the last 6-month record for each child, for each episode.  
select if Flag\_LastRpt4Ep=1.  
execute.

delete variables Flag\_LastRpt4Ep.

\* Select only children who were served during the 12-month period specified earlier.

\*\*\*\*\*

\* This is the cohort of children for whom performance will be examined.

\* Flag the records to keep.  
if ((DtLatRem lt DtPeriodBeg) and (DtDisch ge (DtPeriodBeg) | missing(DtDisch))) InAtStart = 1.  
if ((DtLatRem ge DtPeriodBeg) and (DtLatRem le DtPeriodEnd)) Entered = 1.  
if (InAtStart = 1 or Entered = 1) Served = 1.  
recode InAtStart Entered (sysmis = 0).  
execute.

variable labels  
InAtStart 'Child was in care on the first day of the 12-month period'  
Entered 'Child entered care during the specified 12-month period'.  
value labels InAtStart Entered  
0 'No'  
1 'Yes'.  
frequencies Entered InAtStart.

\* Select the records to keep.  
select if Served = 1.  
execute.

\* Handling multiple episodes in the 12-month period.

\*\*\*\*\*

\* For maltreatment in foster care, all of a child's episodes during the 12-month period are considered.  
\* For example, if a child has two entries in the 12-month period, data from both episodes are used.

\*\*\*\*\*

\* REMOVE RECORDS WITH DQ PROBLEMS RELEVANT TO THIS MEASURE.

\*\*\*\*\*

\* Do not delete dropped records if they occur in the last period needed to calculate observed performance.

\* This ensures we use data only from submissions required to observe the cohort.

if DtReportEnd eq DtPeriodEnd DQ\_Dropped = 0.

execute.

\* Flag records with a problem (i.e., = 1) for any of the AFCARS DQ checks used for Maltreatment in foster care,

\* except DQ\_IDNoMatchNext6Mo and DQ\_totalrem1. The NCANDS DQ checks will be applied later.

compute DQ\_Indicator=0.

if any(1,DQ\_DOBgtDtDisch, DQ\_DOBgtDtLatRem, DQ\_Dropped, DQ\_DtDischeqDtLatRem, DQ\_DtDischlDtLatRem, DQ\_gt21DOBtoDtDisch, DQ\_gt21DOBtoDtLatRem, DQ\_gt21DtDischtoDtLatRem, DQ\_missDOB, DQ\_missDtLatRem) DQ\_Indicator=1.

execute.

\* For each state, report the number and % of child records that will be removed due to AFCARS DQ.

ctables

/VLABELS VARIABLES=stateabb DQ\_Indicator DISPLAY=LABEL

/TABLE stateabb [COUNT F40.0, ROWPCT.COUNT PCT40.1] BY DQ\_Indicator

/CATEGORIES VARIABLES=stateabb DQ\_Indicator ORDER=A KEY=VALUE EMPTY=INCLUDE.

\* For each state, report the number of cases with a problem, by AFCARS DQ check.

temporary.

select if DQ\_Indicator = 1.

ctables

/VLABELS VARIABLES=stateabb DISPLAY=NONE /VLABELS VARIABLES=DQ\_Indicator DQ\_DOBgtDtDisch DQ\_DOBgtDtLatRem

DQ\_Dropped DQ\_DtDischeqDtLatRem DQ\_DtDischlDtLatRem DQ\_gt21DOBtoDtDisch DQ\_gt21DOBtoDtLatRem

DQ\_gt21DtDischtoDtLatRem DQ\_missDOB DQ\_missDtLatRem

DISPLAY=LABEL

/TABLE stateabb [C][COUNT F40.0] BY DQ\_Indicator [C] + DQ\_DOBgtDtDisch [C] + DQ\_DOBgtDtLatRem [C] +

DQ\_Dropped [C] + DQ\_DtDischeqDtLatRem [C] + DQ\_DtDischlDtLatRem [C] + DQ\_gt21DOBtoDtDisch [C] +

DQ\_gt21DOBtoDtLatRem [C] + DQ\_gt21DtDischtoDtLatRem [C] + DQ\_missDOB [C] +

DQ\_missDtLatRem [C]

/SLABELS VISIBLE=NO

/CATEGORIES VARIABLES=stateabb DQ\_Indicator ORDER=A KEY=VALUE EMPTY=EXCLUDE

/CATEGORIES VARIABLES=DQ\_DOBgtDtDisch [1] EMPTY=INCLUDE

/CATEGORIES VARIABLES=DQ\_DOBgtDtLatRem [1] EMPTY=INCLUDE

/CATEGORIES VARIABLES=DQ\_Dropped [1] EMPTY=INCLUDE

/CATEGORIES VARIABLES=DQ\_DtDischeqDtLatRem [1] EMPTY=INCLUDE

/CATEGORIES VARIABLES=DQ\_DtDischlDtLatRem [1] EMPTY=INCLUDE

/CATEGORIES VARIABLES=DQ\_gt21DOBtoDtDisch [1] EMPTY=INCLUDE

/CATEGORIES VARIABLES=DQ\_gt21DOBtoDtLatRem [1] EMPTY=INCLUDE

/CATEGORIES VARIABLES=DQ\_gt21DtDischtoDtLatRem [1] EMPTY=INCLUDE

```
/CATEGORIES VARIABLES=DQ_missDOB [1] EMPTY=INCLUDE
/CATEGORIES VARIABLES=DQ_missDtLatRem [1] EMPTY=INCLUDE.
```

```
* Delete records with a DQ problem(s).
select if DQ_Indicator=0.
execute.
```

```
* Delete the DQ variables as they are no longer needed.
delete variables DQ_Dropped to DQ_totalrem1 DQ_Indicator.
```

```
*****
```

```
* ADJUST DISCHARGE DATES FOR YOUTH WHO TURN 18
```

```
*****
```

```
* Youth who turn 18 during the 12-month period will not have time in care beyond their 18th birthday or
* victimizations after their 18th birthday counted. To handle this, for children who turn 18 during the period we
* replace their discharge date with the date they turned 18. Therefore, the LOS for children who turn 18 during
* the period will be calculated from date of latest removal to date of 18th birthday.
```

```
*****
```

```
* Calculate child's 18th birthday.
compute Bday18=DATESUM(DtBirth, 18, "years", 'closest').
variable labels Bday18 "18th birthday".
variable level Bday18(SCALE).
formats Bday18(ADATE10).
variable width Bday18(10).
execute.
```

```
* If 18th birthday occurred during the 12-month period, and child had not discharged by the time he turned 18,
* replace date of discharge with date of 18th birthday.
do if (Bday18 gt DtPeriodBeg and Bday18 le DtPeriodEnd) and (missing(DtDisch) or DtDisch gt Bday18).
compute DtDischAdjusted=Bday18.
compute Adjust18=1.
end if.
execute.
```

```
variable labels DtDischAdjusted 'For children who turned 18 during the period and had not discharged by that time, the date of their 18th birthday is their effective date of discharge'.
```

```
* Copy date of discharge for remaining children to DtDischAdjusted.
do if sysmis(DtDischAdjusted).
compute DtDischAdjusted=DtDisch.
end if.
execute.
formats DtDischAdjusted (adate10).
```

```
*****
```

\* CALCULATE AGE ON FIRST DAY

\*\*\*\*\*

\* If child entered during the 12-month period, we use age entry (already calculated and part of source data).

\* If the child was in care on the first day of the 12-month period, we use age on first day.

\* Age on first day (categorical), specified as 0-3 mos, 4-11 mos, 1-5 yrs, 6-10 yrs, 11-16 yrs, 17, 18-21 yrs.

\* Useful for reporting and descriptive statistics.

\*\*\*\*\*

\* Calculate age on first day in months.

compute AgeFDmos = datediff(DtPeriodBeg,DtBirth,"months").

\* Record age in months to desired categories.

recode AgeFDmos (216 thru 263=6) (204 thru 216=5) (132 thru 204=4) (72 thru 132=3) (12 thru 72=2) (4 thru 12=1) (0 thru 4=0) (else=sysmis) into AgeFDmosyrsCat.

execute.

\* Age on first day (interval), specified as 0-3 mos, 4-11 mos, 1 yr ... 21 yrs.

\* Used later in risk adjustment, where each age value is converted to a yes/no (1/0) dummy variable.

\*\*\*\*\*

\* Calculate age on first day in years.

compute AgeFDyrs = datediff(DtPeriodBeg,DtBirth,"years").

\* Children < 1 currently have a value of 0 for AgeFDyrs. Need to split these into 0-3 mos (0) mos and 4-11 mos (1).

\* To make room for the 0 and 1 for < 1 children, make a copy of AgeFDyrs then shift all subsequent ages, so age1 = 2, age2 = 3, etc.

compute AgeFDmosyrs = AgeFDyrs.

\* Recode 1 - 21 to 2 - 22.

recode AgeFDyrs (1=2) (2=3) (3=4) (4=5) (5=6) (6=7) (7=8) (8=9) (9=10) (10=11) (11=12) (12=13) (13=14) (14=15) (15=16) (16=17) (17=18) (18=19) (19=20) (20=21) (21=22) INTO

AgeFDmosyrs.

\* If child is 0-3, make AgeFDmosyrs = 0.

\* If child is 4-11, make AgeFDmosyrs = 1.

if AgeFDmosyrsCat = 0 AgeFDmosyrs = 0.

if AgeFDmosyrsCat = 1 AgeFDmosyrs = 1.

execute.

\* Handle problem values.

\*\*\*\*\*

\* If date of birth is after the date of first day (i.e., age on FS is negative), recode to 8888.

\* Note: Negative age on first day is to be expected for children who entered during the 12-month period at age

\* < 1. For these children, their age at entry will be selected so it's okay to assign their age on first day to 8888.

do if DtBirth > DtPeriodBeg.

recode AgeFDmos AgeFDyrs AgeFDmosyrsCat AgeFDmosyrs (else = 8888).

end if.

\* If time between date of birth and date of first day is > 21 yrs (i.e., age on FD is > 21 yrs), recode to 8888.

compute #AgeFD = datediff(DtPeriodBeg,DtBirth,"years").

```
do if #AgeFD > 21.  
recode AgeFDmos AgeFDyrs AgeFDmosyrsCat AgeFDmosyrs (else=8888).  
end if.  
execute.
```

variable labels

```
AgeFDmos 'Age on first day in months'  
AgeFDyrs 'Age on first day in years (Under 1, 1, 2, 3 ... 21)'  
AgeFDmosyrs 'Age on first day in months (0-3 mo, 4-11 mo) and years (1, 2, 3 ... 21)'  
AgeFDmosyrsCat 'Age on first day in months (0-3 mo, 4-11 mo) and years in categories (1-5, 6-10, etc...)'
```

\* Create macro that holds age values.

```
DEFINE !AgeYrs ( )
```

```
0 '< 1 yr'
```

```
1 '1 yr'
```

```
2 '2 yrs'
```

```
3 '3 yrs'
```

```
4 '4 yrs'
```

```
5 '5 yrs'
```

```
6 '6 yrs'
```

```
7 '7 yrs'
```

```
8 '8 yrs'
```

```
9 '9 yrs'
```

```
10 '10 yrs'
```

```
11 '11 yrs'
```

```
12 '12 yrs'
```

```
13 '13 yrs'
```

```
14 '14 yrs'
```

```
15 '15 yrs'
```

```
16 '16 yrs'
```

```
17 '17 yrs'
```

```
18 '18 yrs'
```

```
19 '19 yrs'
```

```
20 '20 yrs'
```

```
21 '21 yrs'
```

```
!ENDDDEFINE.
```

```
DEFINE !AgeMosYrs ( )
```

```
0 '0-3 mos'
```

```
1 '4-11 mos'
```

```
2 '1 yr'
```

```
3 '2 yrs'
```

```
4 '3 yrs'
```

```
5 '4 yrs'
```

```
6 '5 yrs'
```

7 '6 yrs'  
8 '7 yrs'  
9 '8 yrs'  
10 '9 yrs'  
11 '10 yrs'  
12 '11 yrs'  
13 '12 yrs'  
14 '13 yrs'  
15 '14 yrs'  
16 '15 yrs'  
17 '16 yrs'  
18 '17 yrs'  
19 '18 yrs'  
20 '19 yrs'  
21 '20 yrs'  
22 '21 yrs'  
!ENDDFINE.

DEFINE !AgeMosYrsCat ( )

0 '0-3 mos'  
1 '4-11 mos'  
2 '1-5 yrs'  
3 '6-10 yrs'  
4 '11-16 yrs'  
5 '17'  
6 '18-21 yrs'  
!ENDDFINE.

\* Age on first day.

value labels

AgeFDmos

8888 'Invalid (age is negative or > 21)'

9999 'Cannot calculate (DOB missing)'

/AgeFDyrs

!AgeYrs

8888 'Invalid (age is negative or > 21)'

9999 'Cannot calculate (DOB missing)'

/AgeFDmosyrs

!AgeMosYrs

8888 'Invalid (age is negative or > 21)'

9999 'Cannot calculate (DOB missing)'

/AgeFDmosyrsCat

!AgeMosYrsCat

8888 'Invalid (age is negative or > 21)'

9999 'Cannot calculate (DOB missing)'.

formats AgeFDmos to AgeFDmosyrs (F2.0).

\*\*\*\*\*

\* SET CHILD'S AGE AS AGE AT ENTRY OR AGE ON FIRST DAY

\*\*\*\*\*

\* If child entered during the 12-month period, set age to age at entry.

\* If child was in care on the first day of the 12-month period, set age to age on first day.

do if Entered=1.

compute Agemos=AgeNmos.

compute AgemosyrsCat=AgeNmosyrsCat.

compute Ageyrs=AgeNyrs.

compute Agemosyrs=AgeNmosyrs.

end if.

do if InAtStart=1.

compute Agemos=AgeFDmos.

compute AgemosyrsCat=AgeFDmosyrsCat.

compute Ageyrs=AgeFDyrs.

compute Agemosyrs=AgeFDmosyrs.

end if.

execute.

variable labels

Agemos 'Age on first day (or at entry) in months'

Ageyrs 'Age on first day (or at entry) in years (Under 1, 1, 2, 3 ... 21)'

Agemosyrs 'Age on first day (or at entry) in months (0-3 mo, 4-11 mo) and years (1, 2, 3 ... 21)'

AgeNmosyrsCat 'Age on first (or at entry) in months (0-3 mo, 4-11 mo) and years in categories (1-5, 6-10, etc...)'

value labels

Agemos

8888 'Invalid (age is negative or > 21)'

/Ageyrs

!AgeYrs

8888 'Invalid (age is negative or > 21)'

/Agemosyrs

!AgeMosYrs

8888 'Invalid (age is negative or > 21)'

/AgemosyrsCat

!AgeMosYrsCat

8888 'Invalid (age is negative or > 21)'

formats Agemos to Agemosyrs (F2.0).

\*\*\*\*\*

\* CALCULATE LENGTH OF STAY

\*\*\*\*\*

\* LOS calculations differ based on when the children was in care (i.e., on the first day of or entered during the  
\* 12-month period) and when the child exited (i.e., during the 12-month period or after, if at all).

\* We already have a flag to indicate if the child was InAtStart or Entered. Now create a flag to indicate if the  
\* child exited during the 12-month period.

if ((DtDisch ge DtPeriodBeg) and (DtDisch le DtPeriodEnd)) Exited = 1.

recode Exited (sysmis = 0).

execute.

variable labels Exited 'Child exited care during the specified 12-month period'.

value labels Exited

0 'No'

1 'Yes'.

frequencies Exited.

compute LOSdays = \$sysmis.

\* If child entered and exited during the 12-month period, LOS is time between DtLatRem and DtDischAdjusted.

do if Entered=1 and Exited=1.

compute LOSdays=datediff(DtDischAdjusted,DtLatRem,"days").

end if.

\* If child was in care at the start of the 12-month period, and exited during the 12-month period, LOS is time

\* between DtPeriodBeg and DtDischAdjusted.

do if InAtStart=1 and Exited=1.

compute LOSdays=datediff(DtDischAdjusted,DtPeriodBeg,"days").

end if.

\* If child entered during the 12-month period, and did not exit during the 12-month period, LOS is time between

\* DtLatRem to DtPeriodEnd.

do if Entered=1 and Exited=0.

compute LOSdays=datediff(DtPeriodEnd,DtLatRem,"days").

end if.

\* If child was in care at the start of the 12-month period, and did not exit during the 12-month period, LOS is

\* time between DtPeriodBeg and DtPeriodEnd. These children were in care continuously across the entire year,

\* so we add a day to their LOS. Otherwise, SPSS does not count inclusive of the start and end dates.

do if InAtStart=1 and Exited=0.

compute LOSdays=datediff(DtPeriodEnd,DtPeriodBeg,"days").

compute LOSdays=LOSdays+1.

end if.

execute.

\* At this point, several children will have negative LOSs. These are children who entered during the period at age

\* age 18 or older. They will be excluded later.

\* frequencies LOSdays.

variable labels

LOSdays 'LOS - Days in care during the 12-month period'.

formats LOSdays (F4.0).

\*\*\*\*\*

\* REMOVE EPISODES THAT MEET EXCLUSION CRITERIA

\*\*\*\*\*

\* The denominator for this indicator is the sum of children's LOS (days) across \*all\* episodes during the  
\* 12-month period. So if a child entered twice during the 12-month periods, his total LOS is the LOS from his  
\* first episode plus the LOS from his second episode. However, when summing a child's LOS over all his  
\* episodes in the 12-month period, we want to exclude:  
\* 1) episodes in which the child entered at age 18 or more and  
\* 2) \*complete\* episodes with LOS < 8 days. If the episode has a LOS < 8 days, but the child is still in care,  
\* we want to keep this. These short but ongoing episodes represent entries that occurred near the end of the  
\* 12-month period and continued past it (i.e., child was still in care).

\* Complete episodes with LOS < 8 days.

compute ExLOS8 = 0.

if (Entered=1 and Exited=1) and LOSdays lt 8 ExLOS8=1.

do if InAtStart=1 and Exited=1.

compute TempLOS8= datediff(DtDischAdjusted,DtLatRem,"days").

if TempLOS8 lt 8 ExLOS8=1.

end if.

execute.

\* Age at entry or on first day is 18 or older.

compute ExAge18 = 0.

if AgemosyrsCat = 6 ExAge18 = 1.

execute.

\* Age at entry or on first day is invalid (negative or > 21).

if AgemosyrsCat = 8888 ExAge18 = 8888.

execute.

variable labels

ExLOS8 'Episode has a LOS < 8 days'

ExAge18 'Child was age 18 or older at entry or on first day'.

value labels ExLOS8

0 'No'

1 'Yes'

9999 'Cannot be determined because LOS is negative or > 21'

/ ExAge18

0 'No'

1 'Yes'  
8888 'Cannot be determined because age is negative or > 21'.  
execute.

formats ExLOS8 to ExAge18 (F1.0).

\* Report number and % of episodes that will be excluded, by state.

```
crosstabs  
  /TABLES=stateabb BY ExLOS8 ExAge18  
  /FORMAT=AVALUE TABLES  
  /CELLS=COUNT  
  /COUNT ROUND CELL.
```

\* Select the records to keep (i.e., 0 for both vars).  
select if (ExLOS8 = 0 and ExAge18 = 0).  
execute.

```
*****  
* SUM EACH CHILD's LOS ACROSS EPISODES  
*****
```

```
sort cases BY ChildID.  
aggregate  
  /OUTFILE=* MODE=ADDVARIABLES  
  /PRESORTED  
  /BREAK=ChildID  
  /Den_Child=SUM(LOSdays).
```

variable labels Den\_Child 'Maltreatment in care denominator - Childs total length of stay (days) across all episodes in the 12-month period'.  
formats Den\_Child (F3.0).

\* Some children may have a total LOS > 365 days.  
\*\*\*\*\*

\* These are due to data quality issues associated with some children who were reported two or more times  
\* during the 12-month period, with different dates of latest removal, and with no date of discharge in the first  
\* reported episode.

```
* frequencies Den_Child.  
do if Den_Child gt 365.  
compute Den_Child=365.  
end if.  
execute.
```

\* Some children may have a total LOS = 0 days.

\*\*\*\*\*

- \* These are children who exited on the first day of the 12-month period, or entered on the last day of the
- \* 12-month period. At these points, they have not been in care during the 12-month period for a full 24 hours.
- \* In addition, victimization rates for these children cannot be calculated when their LOS (i.e., denominator) is 0.

select if Den\_Child <> 0.  
execute.

\*\*\*\*\*

\* FINAL PREP OF AFCARS DATA

\*\*\*\*\*

- \* Count the number of episodes. Maximum number should be 2 (one for each 6-month reporting period).
- sort cases by ChildID DtLatRem.  
compute numepisodes = 1.  
if (ChildID = lag(ChildID))numepisodes = lag(numepisodes) + 1.  
frequencies numepisodes.

- \* This indicator excludes any maltreatment report that occurs within 7 days of removal from home (DtLatRem).
  - \* Therefore, compute DtLatRemPlus7 to be the removal date plus seven days. This will be used for later to
  - \* exclude maltreatment reports in NCANDS that occurred before DtLatRemPlus7.
- compute DtLatRemPlus7 = DtLatRem +(7\*86400).  
execute.  
formats DtLatRemPlus7 (adate10).  
variable labels DtLatRemPlus7 'Childs date of latest removal plus 7 days'.

- \* Later we need to find only maltreatment reports that occurred after the date of latest removal and before
  - \* the date of discharge. If the the child is still in care at the end of the 12-month period, his DtDisch is missing,
  - \* in which case we use DtPeriodEnd as the effective date of discharge.
- if sysmis(DtDischAdjusted) DtDischAdjusted2=DtPeriodEnd.  
if sysmis(DtDischAdjusted2) DtDischAdjusted2=DtDischAdjusted.  
execute.  
formats DtDischAdjusted2 (adate10).

variable labels DtDischAdjusted2 'Discharge date used for NCANDS matching. For children who did not exit during the 12-month period, DtPeriodEnd is their effective date of discharge'.

- \* Save.
- save outfile= 'Maltx FC 1 AFCARS duplicated ' + PeriodType + YYstr + '.sav'  
/keep = state stateabb statetxt TwelveMoCohort recnumbr ChildID Ageyrs Agemosyrs AgemosyrsCat sex DtLatRemPlus7 DtDischAdjusted2 Den\_Child.  
dataset name AFCARSDuplicated.

- \* Restructure AFCARS data from long to wide (one record per child)

\*\*\*\*\*

- \* Current AFCARS file has one record per episode during the 12-month period, so a child with two episodes
- \* has two records. This new file will contain one record per child, with all of his episode information on one
- \* record, indexed by N. For example: state ChildID recnumbr DtDischAdjusted.1 DtDischAdjusted.2 ... etc.
- \* We will later match this file with NCANDS (matching on recnumbr, which in NCANDS is called afcarsid).
- \* There will never be more than two episodes per child (i.e., variable.2) that are indexed in the AFCARS wide file,
- \* since it is based on two 6-month periods and each period can have only episode.

sort cases BY ChildID DtLatRemPlus7 .

casestovars

/ID = ChildID

/GROUPBY = VARIABLE.

- \* Rename AFCARS variables to match NCANDS variables

\*\*\*\*\*

- \* Later we need to match children in NCANDS with children in AFCARS. We match based on the state and the
- \* child's AFCARS ID. Rename AFCARS variables to confirm with NCANDS variables:
- \* ... For state, NCANDS uses staterr, which is already in the file (when AFCARS source data was created).
- \* ... For AFCARS ID, NCANDS uses afcarsid, AFCARS uses recnumbr.

\* string staterr (a6).

\* compute staterr=stateabb.

string afcarsid (a36).

compute afcarsid= recnumbr.

execute.

- \* Flag records indicating they are from AFCARS

\*\*\*\*\*

- \* This will be used for denominator for determining the number of AFCARS matches in the NCANDS file.

compute AFCARS = 1.

frequencies AFCARS.

- \* Save.

sort cases by staterr afcarsid.

save outfile ='Maltx FC 2 AFCARS wide ' + PeriodType + YYstr + '.sav'.

\*\*\*\*\*

\*\*\*\*\*

- \* PREPARE NCANDS DATA

\*\*\*\*\*

\*\*\*\*\*

get file 'Submissions NCANDS\AllState' + YYstr + 'ChildFiles.sav'.

dataset name NCANDS window=front.  
dataset close AFCARSDuplicated.

\* Prevent accidentally overwriting source data.  
dataset copy NCANDSChild.  
dataset activate NCANDSChild.  
dataset close NCANDS.

\* Delete some unused variables to make file more manageable.  
delete variables  
chracai TO cethn  
chlvng TO chmil  
cdalc TO per3mal4.

\* Flag victims, where a victim is a child who died due to maltreatment (maldeath = 1) or has a disposition of  
\* substantiated (malNlevel = 1) or indicated (malNlevel = 2), for any of four possible maltreatments.  
compute flvictim = 0.  
if(mal1lev le 2)flvictim = 1.  
if(mal2lev le 2 )flvictim = 1.  
if (mal3lev le 2)flvictim = 1.  
if (mal4lev le 2)flvictim = 1.  
if (maldeath = 1)flvictim = 1.

\* Select only victims.  
select if flvictim = 1.  
execute.

\*\*\*\*\*  
\* FLAG STATES THAT EXCEED DQ LIMITS ON NCANDS WITHIN-FILE CHECKS  
\*\*\*\*\*

\* The cross-file check will be handled later (i.e., Some victims with AFCARS IDs should match IDs in  
\* AFCARS file).

\* Flag state if more than 5% of victims have a missing age.  
\*\*\*\*\*

frequencies chage.  
compute hasage = 0.  
if (chage ge 0 and chage lt 99) hasage = 1.  
variable labels hasage 'Child age is reported'.  
value labels hasage  
0 'No'  
1 'Yes'.  
execute.

\* Create file holding percentage of cases with missing age by state and whether state failed the DQ check.  
\* This file will be merged in toward the end of the syntax when we are ready to review and remove states that  
\* failed the check.

oms select tables

```
/ destination format = sav outfile = 'DQ NCANDS\NCANDS_DQ_MissingAge ' + PeriodType + YYstr + '.sav'  
/ if commands = ['crosstabs'] subtypes = ['Crosstabulation'].  
crosstabs /TABLES=staterr BY hasage /FORMAT=AVALUE TABLES /CELLS=ROW /COUNT ROUND CELL.  
omsend.
```

```
get file 'DQ NCANDS\NCANDS_DQ_MissingAge ' + PeriodType + YYstr + '.sav'.
```

```
dataset name NCANDS_DQ_MissingAge window=front.
```

\* states with 0 missing ages have blanks for No; replace with 0.0%.

\* delete last "Total" row.

\* rename Var2 to staterr and make it 6 wide.

```
if sysmis(No) No = 0.0.
```

```
select if not (Var1 = "Total").
```

```
rename variables (Var2 No = staterr MissingAge).
```

```
alter type staterr(A2=A6).
```

```
if MissingAge > 5 DQFailedCheck_Age = 1.
```

```
if sysmis(DQFailedCheck_Age) DQFailedCheck_Age=0.
```

```
sort cases by staterr.
```

```
execute.
```

```
save outfile = 'DQ NCANDS\NCANDS_DQ_MissingAge ' + PeriodType + YYstr + '.sav'
```

```
/keep staterr MissingAge DQFailedCheck_Age.
```

```
dataset activate NCANDSChild.
```

```
dataset close NCANDS_DQ_MissingAge.
```

\* Merge in results from NCANDS DQ check to flag states that failed the check.

```
match files
```

```
/FILE=*
```

```
/TABLE='DQ NCANDS\NCANDS_DQ_MissingAge ' + PeriodType + YYstr + '.sav'
```

```
/BY staterr
```

```
/KEEP cfiledat to hasage DQFailedCheck_Age.
```

```
execute.
```

\* Flag state if fewer than 1% of victims have an AFCARS ID.

```
*****
```

\* Count number of NCANDS victims with AFCARS IDs.

```
compute hasafcars = 0.
```

```
if (afcarsid ne ' ') hasafcars = 1.
```

```
variable labels hasafcars 'AFCARS ID is reported'.
```

```
value labels hasafcars
```

0 'No'  
1 'Yes'.  
execute.  
frequencies hasafcars.

\* Create file holding percentage of cases with an AFCARS ID by state and whether state failed the DQ check.  
\* This file will be merged in toward the end of the syntax when we are ready to review and remove states that  
\* failed the check.

oms select tables

```
/ destination format = sav outfile = 'DQ NCANDS\NCANDS_DQ_HasAFCARSID ' + PeriodType + YYstr + '.sav'  
/ if commands = ['crosstabs'] subtypes = ['Crosstabulation'].  
crosstabs /TABLES=staterr BY hasafcars /FORMAT=AVALUE TABLES /CELLS=ROW /COUNT ROUND CELL.  
omsend.
```

```
get file 'DQ NCANDS\NCANDS_DQ_HasAFCARSID ' + PeriodType + YYstr + '.sav'.
```

```
dataset name NCANDS_DQ_AFCARSID window=front.
```

\* states with 0 AFCARS IDs have blanks for Yes; replace with 0.0%.

\* delete last "Total" row.

\* rename Var2 to staterr and make it 6 wide.

```
if sysmis(Yes) Yes = 0.0.
```

```
select if not (Var1 = "Total").
```

```
rename variables (Var2 Yes = staterr HasAFCARSID).
```

```
alter type staterr(A2=A6).
```

```
if HasAFCARSID < 1 DQFailedCheck_HasID = 1.
```

```
if sysmis(DQFailedCheck_HasID) DQFailedCheck_HasID=0.
```

```
sort cases by staterr.
```

execute.

```
save outfile = 'DQ NCANDS\NCANDS_DQ_HasAFCARSID ' + PeriodType + YYstr + '.sav'
```

```
/keep staterr HasAFCARSID DQFailedCheck_HasID.
```

```
dataset activate NCANDSChild.
```

```
dataset close NCANDS_DQ_AFCARSID.
```

\* Merge in results from NCANDS DQ check to identify states that failed the check.

match files

```
/FILE=*  
/TABLE='DQ NCANDS\NCANDS_DQ_HasAFCARSID ' + PeriodType + YYstr + '.sav'  
/BY staterr  
/KEEP cfiledat to hasafcars DQFailedCheck_HasID.
```

execute.

\*\*\*\*\*

\* REMOVE STATES THAT EXCEED DQ LIMITS ON ANY OF THE WITHIN-FILE CHECKS

\*\*\*\*\*

\* Review and verify states that will be excluded.

\*\*\*\*\*

temporary.

select if DQFailedCheck\_Age = 1 OR DQFailedCheck\_HasID = 1.

ctables

/VLABELS VARIABLES=staterr DISPLAY=NONE /VLABELS VARIABLES=DQFailedCheck\_Age DQFailedCheck\_HasID

DISPLAY=LABEL

/TABLE staterr [C][COUNT F40.0] BY DQFailedCheck\_Age [C] + DQFailedCheck\_HasID [C]

/SLABELS VISIBLE=NO

/CATEGORIES VARIABLES=staterr ORDER=A KEY=VALUE EMPTY=EXCLUDE

/CATEGORIES VARIABLES=DQFailedCheck\_Age [1] EMPTY=INCLUDE

/CATEGORIES VARIABLES=DQFailedCheck\_HasID [1] EMPTY=INCLUDE.

\* Select only states that met the DQ checks.

select if DQFailedCheck\_Age = 0 AND DQFailedCheck\_HasID = 0.

execute.

delete variables DQFailedCheck\_Age DQFailedCheck\_HasID.

\* Count the number of states remaining.

compute numstates = 1.

if (staterr eq lag(staterr))numstates = 0.

frequencies numstates.

\*\*\*\*\*

\* SELECT APPLICABLE RECORDS

\*\*\*\*\*

\* If child has a subsequent report, and it is within 1 day of his initial report, we do not count it (i.e, reports are

\* treated as 'rolled-up' into a single report). Note: 86,400 is number of seconds in a day and is measurement

\* used in SPSS date fields.

sort cases by staterr chid rptdt.

compute daysdiff = (rptdt - lag(rptdt))/86400.

select if (chid ne lag(chid) or daysdiff gt 1).

\* Select only reports with dates occurring during the 12-month period.

select if (rptdt ge date.mdy(10,01,YYYY-1)).

execute.

\* Keep only records with an afcarsid.

select if (hasafcars = 1).

freq hasafcars.

\* Remove children age 18 and older.

select if chage lt 18.

execute.

\*\*\*\*\*

\* MERGE EPISODE DATA FROM AFCARS  
INTO NCANDS TO IDENTIFY CHILDREN VICTIMIZED WHILE IN CARE

\*\*\*\*\*

\* Merge fields from AFCARS wide file into NCANDS file. NCANDS file includes one record for every report a  
\* child may have had; AFCARS file includes one record per child.

sort cases by staterr afcarsid.

match files

/file = \*

/table 'Maltx FC 2 AFCARS wide ' + PeriodType + YYstr + '.sav'

/by staterr afcarsid.

execute.

\* Identify children victimized while in care for any of his episodes (up to two) in the 12-month period.

\* At this point, the count of victimInCare will be duplicate since a child has a record for every maltreatment report

\* during the 12-month period. When we aggregate the file later and select max(victimInCare), we will have a

\* unique count of children who were a victim in care at any point during the 12-month period.

compute victimInCare = 0.

if (rptdt ge DtLatRemPlus7.1 and rptdt lt DtDischAdjusted2.1)victimInCare = 1.

if (rptdt ge DtLatRemPlus7.2 and rptdt lt DtDischAdjusted2.2)victimInCare = 1.

\* Count victimizations. For example, if a child has three reports in NCANDS (and is a victim in each report), and

\* all three reports were reported (or the incident occurred) during the child's first episode (DtLatRemPlus7.1

\* and DtDischAdjusted2.1), victimization1 will = 1 for all three records/reports. When we aggregate the file later

\* and sum the victimizations (e.g., sum(victimization1), the result will show 3 victimizations for that one episode.

compute victimization1 = 0.

compute victimization2 = 0.

if (rptdt ge DtLatRemPlus7.1 and rptdt lt DtDischAdjusted2.1)victimization1 = 1.

if (rptdt ge DtLatRemPlus7.2 and rptdt lt DtDischAdjusted2.2)victimization2 = 1.

freq victimization1 victimization2.

\* Incident date adjustment. Recode victimInCare to 0 and victimization.N to 0 if the incident date is present and

\* shows the child was not victimized while in care (i.e., the actual date of maltreatment was before date of latest

\* removal plus 7, or after (or the same day of) the date of discharge.

do if ((victimization1=1) and not(missing(inciddt)) and ((inciddt lt DtLatRemPlus7.1) or (inciddt ge DtDischAdjusted2.1))).

compute adjusted=1.

compute adjustvictim1=1.

compute victimization1=0.

compute victimIncare=0.

end if.

execute.

do if ((victimization2=1) and not(missing(inciddt)) and ((inciddt lt DtLatRemPlus7.2) or (inciddt ge DtDischAdjusted2.2))).

compute adjusted=1.

```
compute adjustvictim2=1.
compute victimIncare=0.
compute victimization2=0.
end if.
execute.
```

```
crosstabs staterr by adjusted.
frequencies adjustvictim1 adjustvictim2.
```

\* Restructure data from long to wide (one record per child)

\*\*\*\*\*

\* Current file has one record per maltreatment report during the 12-month period, so a child with three reports has three records. This new file will contain one record per child, with summary data from all of his maltreatment reports on one record. It will contain only victims in care during the 12-month period; it will not contain children in care who were not victims. The file will hold variables indicating if the NCANDS child has an AFCARSID in the NCANDS file (hasafcars = 1), also exists in the AFCARS file (AFCARS = 1), was a victim in care at any point during the 12-month period (victimInCare = 1), and how many victimizations he had for each of up to two possible foster care episodes (victimization1 and victimization2). "AFCARS" is the only variables from the AFCARS file that is kept.

```
dataset declare VictimsInCare.
aggregate outfile = 'VictimsInCare'
  /break = staterr afcarsid
  /FFY 'Federal Fiscal Year' = first (subyr)
  /NCANDSvic 'Unique NCANDS victim with AFCARS ID' = max(hasafcars)
  /AFCARSMatch 'Unique NCANDS victims with AFCARS matches' = max(AFCARS)
  /victimInCare 'Unique count of children victimized while in care' = max(victimInCare)
  /victimization1 'victim report1' = sum(victimization1)
  /victimization2 'victim report2' = sum(victimization2).
```

```
dataset activate VictimsInCare.
value labels AFCARSMatch
0 'No'
1 'Yes'.
```

```
dataset close NCANDSChild.
```

\* Total up all victimizations for each child.

```
compute Num_Child = sum(victimization1 to victimization2).
frequencies Num_Child victimincare.
variable labels Num_Child 'Number of victimizations child experienced while in care across all episodes n the 12-month period'.
```

\* Save.

```
save outfile = 'Maltx FC 3 Unique victims in care ' + YYstr + '.sav'.
```

\*\*\*\*\*

\* MERGE VICTIMS IN CARE FROM NCANDS INTO AFCARS

\*\*\*\*\*

```
get file = 'Maltx FC 2 AFCARS wide ' + PeriodType + YYstr + '.sav'.
dataset name Unduplicated window=front.
dataset close VictimsInCare.
```

\* Merge data for victims in care during the 12-month period into the AFCARS file, which contains all children  
\* served in the 12-month period, but no information about their victim status or presence/absence in NCANDS.  
\* This will merge in the following fields: FFY, NCANDSVic, AFCARSMatch, victimInCare, victimization1,  
\* victimization2.

```
sort cases by staterr afcarsid.
```

```
match files
```

```
  /file = *  
  /file = 'Maltx FC 3 Unique victims in care ' + YYstr + '.sav'  
  /by staterr afcarsid.
```

```
execute.
```

\* File now includes children from NCANDS that were not in the AFCARS file (presumably because they did not  
\* enter foster care and were never reported in AFCARS). These cases have blanks for all their AFCARS variables.  
\* We want to get rid of these records; delete them.

```
select if (Den_Child ge 0).
```

```
execute.
```

\* File now contains all children in AFCARS served during the 12-month period. If the child was also in NCANDS,  
\* we have data about whether the child was a victim while in care during the period (victimInCare) and the number  
\* of victimizations he experienced during his time in care (Num\_Child). If the child was not in NCANDS, these  
\* data are currently blank, and we assume - because the child was not reported in NCANDS - that he is not a  
\* victim. Populate their data accordingly to show they were in AFCARS but not in NCANDS (AFCARSMatch = 0),  
\* were not an NCANDS victim (NCANDSVic = 0), were not a victimincare (victimInCare = 99), and had zero  
\* victimizations (Num\_Child = 0).

```
do if missing(FFY).
```

```
compute FFY=2013.
```

```
compute NCANDSVic=0.
```

```
compute AFCARSMatch=0.
```

```
compute victimincare=99.
```

```
compute Num_Child=0.
```

```
end if.
```

```
execute.
```

```
missing values victimInCare (99).
```

\*\*\*\*\*

\* FLAG STATES THAT EXCEED DQ LIMITS ON NCANDS CROSS-FILE CHECK

\*\*\*\*\*

\* States failing the within-file NCANDS checks have already been removed. This is the last check to consider.

\* For states that report an AFCARS ID in NCANDS, flag the states for which 0 children in AFCARS match  
\* on that same AFCARS ID.

\*\*\*\*\*

\* Create file holding percentage of victims with an AFCARS ID (reported in NCANDS) with no match in  
\* AFCARS file.

oms select tables

  / destination format = sav outfile = 'DQ NCANDS\NCANDS\_DQ\_AFCARSMatch ' + PeriodType + YYstr + '.sav'

  / if commands = ['crosstabs'] subtypes = ['Crosstabulation'].

crosstabs /TABLES=staterr BY AFCARSMatch /FORMAT=AVALUE TABLES /CELLS=ROW /COUNT ROUND CELL.

omsend.

get file 'DQ NCANDS\NCANDS\_DQ\_AFCARSMatch ' + PeriodType + YYstr + '.sav'.

dataset name NCANDS\_DQ\_AFCARSMatch window=front.

\* states with 0 matches have blanks for Yes; replace with 0.0%.

\* delete last "Total" row.

\* rename Var2 to staterr and make it 6 wide.

if sysmis(Yes) Yes = 0.0.

select if not (Var1 = "Total").

rename variables (Var2 Yes = staterr AFCARSMatch).

alter type staterr(A2=A6).

if AFCARSMatch = 0.0 DQFailedCheck\_AFCARSMatch = 1.

if sysmis(DQFailedCheck\_AFCARSMatch) DQFailedCheck\_AFCARSMatch=0.

sort cases by staterr.

execute.

save outfile = 'DQ NCANDS\NCANDS\_DQ\_AFCARSMatch ' + PeriodType + YYstr + '.sav'

  /keep staterr AFCARSMatch DQFailedCheck\_AFCARSMatch.

dataset activate Unduplicated.

dataset close NCANDS\_DQ\_AFCARSMatch.

\* Merge in results from NCANDS DQ check to identify states that failed the check.

match files

  /FILE=\*

  /TABLE='DQ NCANDS\NCANDS\_DQ\_AFCARSMatch ' + PeriodType + YYstr + '.sav'

  /BY staterr

  /KEEP ChildID to Num\_Child DQFailedCheck\_AFCARSMatch.

execute.

\*\*\*\*\*

\* REMOVE STATES THAT EXCEED DQ LIMITS ON THE CROSS-FILE CHECK

\*\*\*\*\*

\* Review and verify states that will be excluded.

\*\*\*\*\*

temporary.

select if DQFailedCheck\_AFCARSMatch = 1.

ctables

/VLABELS VARIABLES=staterr DISPLAY=NONE /VLABELS VARIABLES=DQFailedCheck\_AFCARSMatch

DISPLAY=LABEL

/TABLE staterr [C][COUNT F40.0] BY DQFailedCheck\_AFCARSMatch [C]

/SLABELS VISIBLE=NO

/CATEGORIES VARIABLES=staterr ORDER=A KEY=VALUE EMPTY=EXCLUDE

/CATEGORIES VARIABLES=DQFailedCheck\_AFCARSMatch [1] EMPTY=INCLUDE.

\* Select only states that met the DQ checks.

select if DQFailedCheck\_AFCARSMatch = 0.

execute.

delete variables DQFailedCheck\_AFCARSMatch.

\* Count the number of states remaining.

compute numstates = 1.

if (staterr eq lag(staterr))numstates = 0.

frequencies numstates.

\*\*\*\*\*

\* SUMMARY STATISTICS

\*\*\*\*\*

\* Create variables holding state and national performance.

\*\*\*\*\*

\* Child-level data.

compute Perf\_Child = (Num\_Child / Den\_Child).

compute Perf\_Child\_MP = (Num\_Child / Den\_Child) \* 365.

execute.

\* State-level data.

sort cases by state (A) ChildID (A).

aggregate

/OUTFILE=\* MODE=ADDVARIABLES

/PRESORTED

/BREAK=state

/Num\_State=SUM(Num\_Child)

/Den\_State=SUM(Den\_Child)

/N\_State=N.

```
compute Perf_State = (Num_State / Den_State).
compute Perf_State_MP = (Num_State / Den_State) * 100000.
execute.
```

\* National-level data.

```
aggregate
  /OUTFILE=* MODE=ADDVARIABLES
  /BREAK=
  /Num_Nation=SUM(Num_Child)
  /Den_Nation=SUM(Den_Child)
  /N_Nation=N.
```

```
compute Perf_Nation = (Num_Nation / Den_Nation).
compute Perf_Nation_MP = (Num_Nation / Den_Nation) * 100000.
execute.
```

```
formats Perf_Child Perf_State Perf_Nation (F6.5).
```

variable labels

```
Perf_Child - Maltx in care - Victimization per 365 days in care (child)
N_State 'Number of children served in the state during the 12-month period'
Den_State 'Maltx in care denominator - Total number days children were in care in state across all episodes in the 12-month period'
Num_State 'Maltx in care numerator - Total number of victimizations in state children experienced while in care across all episodes in the 12-month period'
Perf_State_MP 'Maltx in care - Victimization rate per 100,000 days in care (state)'
N_Nation 'Number of children served in the nation during the 12-month period'
Den_Nation 'Maltx in care denominator - Total number days children were in care in nation across all episodes in the 12-month period'
Num_Nation 'Maltx in care numerator - Total number of victimizations in nation children experienced while in care across all episodes in the 12-month period'
Perf_Nation_MP 'Maltx in care - Victimization rate per 100,000 days in care (nation)'.
```

\* Display observed performance by state and for nation.

ctables

```
/VLABELS VARIABLES=stateabb DISPLAY=NAME /VLABELS VARIABLES=N_State Num_State Den_State
  Perf_State_MP N_Nation Num_Nation Den_Nation Perf_Nation_MP
  DISPLAY=BOTH
/TABLE stateabb [C] BY N_State [S][MAXIMUM] + Num_State [S][MAXIMUM] + Den_State [S][MAXIMUM] +
  Perf_State_MP [S][MAXIMUM] + N_Nation [S][MAXIMUM] + Num_Nation [S][MAXIMUM] + Den_Nation
  [S][MAXIMUM] + Perf_Nation_MP [S][MAXIMUM]
/CATEGORIES VARIABLES=stateabb ORDER=A KEY=VALUE EMPTY=EXCLUDE.
```

\*\*\*\*\*

\* OUTPUT FILES

\*\*\*\*\*

\* Save.

```
save outfile='Performance observed child\CFSR 3 - Observed perf for maltx in care ' + PeriodType + YYstr + '.sav'
  /compressed.
```

```
* Save file for STATA (for multi-level modeling).
* If child has two episodes during the period, use age associated with the first episode.
rename variables (Agesyrs.1 = ChildAge).
save translate OUTFILE='Performance observed child\CFSR 3 - Observed perf for maltx in care ' + PeriodType + YYstr + '.dta'
/TYPE=STATA
/VERSION=8
/EDITION=SE
/MAP
/REPLACE
/KEEP=state stateabb TwelveMoCohort ChildID Num_Child Den_Child ChildAge N_State Den_State
Num_State Perf_State Perf_State_MP N_Nation Den_Nation Num_Nation Perf_Nation Perf_Nation_MP.
rename variables (ChildAge = Agesyrs.1).
```

```
* Create file with one record per state holding observed performance for the 12-month cohort.
compute numstates = 1.
if (state eq lag(state))numstates = 0.
frequencies numstates.
execute.
```

```
dataset copy OneRecordPerState.
dataset activate OneRecordPerState.
select if (numstates=1).
execute.
dataset activate OneRecordPerState.
```

```
string Indicator (A30).
execute.
compute Indicator = "Maltx in care".
execute.
```

```
save outfile='Performance observed state\CFSR 3 - Observed perf for maltx in care State file ' + PeriodType + YYstr + '.sav'
/keep state stateabb statetxt Indicator TwelveMoCohort Perf_State.
```

```
dataset activate Unduplicated.
dataset close OneRecordPerState.
```

```
* Save output.
output save OUTFILE='Output\CFSR 3 - Observed perf for maltx in care ' + PeriodType + YYstr + '.spv'.
```

```
*****
ALL DONE.
*****
```

```
* No need to save anything.
```

new file.

dataset name AllDone WINDOW=FRONT.

dataset close Unduplicated.