

set printback=on.

CFSR 3: OBSERVED PERFORMANCE
- RECURRENCE OF MALTREATMENT.

* 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 files in the 'CFSR3\Submissions NCANDS\' folder:

- * AllStateYYChildFiles.sav
- * 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 two AllStateYYChildFiles.sav file for the FYs needed to measure recurrence of maltreatment for the cohort of interest. For example, if you are interested in examining recurrence of maltreatment for children who were victims during FY12, you will need files called AllState12ChildFiles.sav and AllState13ChildFiles.sav, which contain NCANDS submissions associated with FY12 and FY13, respectively.

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

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

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

SPECIFY THE 12-MONTH COHORT WHOSE OUTCOME WILL BE ASSESSED

- * Identify:
 - * 1. The 12-month cohort of interest (e.g., children who were victimized in FY12) by entering the FY for YYYY and YYstr.
 - * 2. The subsequent 12-month period by entering the subsequent FY (e.g., FY13) for YY2str.

* Note: The FY period spans Oct 1 - Sept 30 of the following year.

* Examples:

- * Children victimized in FY YYYY YYstr YY2str
- * 10 2010 10 11
- * 11 2011 11 12

```
* 12          2012  12  13
* 13          2013  13  14
* etc.
```

```
***** START USER INPUT *****
```

```
define YYYY () 2012
!enddefine.
define YYstr () "12"
!enddefine.
define YY2str () "13"
!enddefine.
```

```
***** END USER INPUT *****
```

```
*****
```

```
GET SOURCE FILES
```

```
*****
```

```
* Determine the subsequent FY based on what the user specified earlier.
```

```
define YYYY2 () YYYY+1
!enddefine.
```

```
* Set working directory.
```

```
cd 'C:\CF SR3'.
show directory.
```

```
* Add two years of child files together.
```

```
add files
  /file = 'Submissions NCANDS\AllState' + YYstr + 'ChildFiles.sav'
  /file = 'Submissions NCANDS\AllState' + YY2str + 'ChildFiles.sav'.
execute.
```

```
dataset name SourceData.
dataset activate SourceData.
```

```
*****
```

```
* FILE PREP
```

```
*****
```

```
* Delete some unused variables to make file more manageable.
```

```
delete variables
chracai TO cethn
chlvng TO chmil
cdalc TO per3mal4.
```

* Verify the records represent the FYs of interest.
frequencies subyr.

* Count the number of states that provided data for each FY.
sort cases BY subyr staterr.
split file LAYERED BY subyr.
compute numstates = 1.
if (staterr eq lag(staterr))numstates = 0.
frequencies numstates.
split file OFF.

* Create variable that indicates the user-specified 12-month cohort.
string TwelveMoCohort (A4).
compute TwelveMoCohort = concat("FY",YYstr).
execute.

* 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.

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

* 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 by FY and whether state failed the DQ check.

* This file will be merged in later to review and remove states that failed the check for either or both years.

oms select tables

/ destination format = sav outfile = 'DQ NCANDS\DQ_MissingAge Recur ' + YYstr + YY2str + '.sav'

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

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

omsend.

get file 'DQ NCANDS\DQ_MissingAge Recur ' + YYstr + YY2str + '.sav'.

dataset name DQ window=front.

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

* delete the "Total" rows.

* convert Var1 (which holds subyr) from string to numeric and rename.

* rename Var2 to staterr and make it 6 wide.

if sysmis(No) No = 0.0.

select if not (Var1 = "Total") and not (Var2 = "Total").

compute subyr = number(Var1, F4.0).

rename variables (Var3 No = staterr MissingAge).

alter type staterr(A2=A6).

execute.

if MissingAge > 5 DQFailedCheck_Age = 1.

if sysmis(DQFailedCheck_Age) DQFailedCheck_Age=0.

sort cases by staterr subyr.

execute.

save outfile = 'DQ NCANDS\DQ_MissingAge Recur ' + YYstr + YY2str + '.sav'

/keep staterr subyr MissingAge DQFailedCheck_Age.

dataset activate SourceData.

dataset close DQ.

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

sort cases BY staterr subyr.

match files

/FILE=*

/TABLE='DQ NCANDS\DQ_MissingAge Recur ' + YYstr + YY2str + '.sav'

/BY staterr subyr

/KEEP cfiledat to hasage DQFailedCheck_Age.

execute.

* If a state failed the within-file check for a given year, it will have a 1 only for that given year.

* Need to assign 1 for the entire state (both years), so the state is excluded later.

aggregate

/OUTFILE=* MODE=ADDVARIABLES

/BREAK=staterr

/DQFailedCheck_Age_EitherYear 'State failed this check in one or both '+

'years'=MAX(DQFailedCheck_Age).

* CROSS-FILE CHECKS - FLAG STATES THAT EXCEED DQ LIMITS ON CROSS-FILE CHECKS

* Preparation

* Recode unborn age 77 to -1 to enable age match later in the syntax.

recode chage (77 = -1).

frequencies chage.

* Create dob and sex variables for each submission year.

do if (subyr = YYYY).

compute chbdate1 = chbdate.

compute chsex1 = chsex.

compute chage1 = chage.

end if.

execute.

do if (subyr = YYYY2).

compute chbdate2 = chbdate.

compute chsex2 = chsex.

compute chage2 = chage.

end if.

execute.

* Count the number of submission years associated with a Child ID.

sort cases by staterr chid subyr.

compute numyears = 1.

if (chid eq lag(chid) and subyr eq lag(subyr)) numyears = 0.

frequencies numyears.

* Determine if child was reported in both years.

dataset declare SourceData_Aggregated.

aggregate outfile = 'SourceData_Aggregated'

 /break = staterr chid

 /numyears = sum(numyears)

 /chbdate1 = max(chbdate1)

 /chsex1 = max(chsex1)

 /chage1 = min(chage1)

 /chbdate2 = max(chbdate2)

 /chsex2 = max(chsex2)

 /chage2 = max(chage2).

dataset activate SourceData_Aggregated.

variable labels numyears 'Child reported in both years'.

value labels numyears

1 'No'

2 'Yes'.

frequencies numyears.

* Flag state if fewer than 1% of victim ChildIDs match across years.

* Create file holding percentage of cases with Child ID reported in both FYs and whether state failed the DQ check.

* This file will be merged in later to review and remove states that failed the check.

oms select tables

 / destination format = sav outfile = 'DQ NCANDS\DQ_ChildIDMatch Recur ' + YYstr + YY2str + '.sav'

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

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

omsend.

get file 'DQ NCANDS\DQ_ChildIDMatch Recur ' + YYstr + YY2str + '.sav'.

dataset name DQ window=front.

* states with 0 matches in both years have blanks for No; replace with 0.0%.

* delete the "Total" rows.

* convert Var1 (which holds subyr) from string to numeric and rename.

* 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 IDMatch).

alter type staterr(A2=A6).

execute.

* If < 1% match, flag as failed.

if IDMatch < 1 DQFailedCheck_IDMatch = 1.

if sysmis(DQFailedCheck_IDMatch) DQFailedCheck_IDMatch=0.

sort cases by staterr.

execute.

save outfile = 'DQ NCANDS\DQ_IDMatch Recur ' + YYstr + YY2str + '.sav'

 /keep staterr IDMatch DQFailedCheck_IDMatch.

dataset activate SourceData.

dataset close DQ.

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

* sort cases BY staterr.

match files

 /FILE=*

 /TABLE='DQ NCANDS\DQ_IDMatch Recur ' + YYstr + YY2str + '.sav'

 /BY staterr

```
/KEEP cfiledat to numyears DQFailedCheck_IDMatch.  
execute.
```

* Flag state if more than 5% of victims ChildIDs match across years, but dates of birth and sex do not match.

```
*****
```

```
dataset activate SourceData_Aggregated.
```

```
compute agediff = chage2-chage1.
```

```
frequencies agediff.
```

* Check to see if dob and sex match when an ID is present in both submission years.

* Note: If state has 0 matches across years, they will fail the previous check. For this current check, agediff and

* samechild will be blank (because there was a not a case in Year 1 and 2 to compare).

```
do if (numyears eq 2 ).
```

```
if ( (chbdate1 eq chbdate2) and (chsex1 eq chsex2))samechild = 1.
```

```
if ( (chbdate1 ne chbdate2) or (chsex1 ne chsex2))samechild = 0.
```

```
if ( (agediff ge 0 and agediff le 3) and (chsex1 eq chsex2))samechild = 1.
```

```
if ( (agediff lt 0 or agediff gt 3) or (chsex1 ne chsex2))samechild = 0.
```

```
end if.
```

```
execute.
```

```
variable labels samechild 'Does DOB and Sex match'.
```

```
value labels samechild
```

```
0 'No'
```

```
1 'Yes'.
```

* Create file holding percentage of cases where Child ID matches but DOB and sex do not and whether state

* failed the DQ check. This file will be merged in later to review and remove states that failed the check.

```
oms select tables
```

```
 / destination format = sav outfile = 'DQ NCANDS\DQ_SameChild Recur ' + YYstr + YY2str + '.sav'
```

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

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

```
omsend.
```

```
get file 'DQ NCANDS\DQ_SameChild Recur ' + YYstr + YY2str + '.sav'.
```

```
dataset name DQ window=front.
```

* states with 0 matches in both years have blanks for No; replace with 0.0%.

* delete the "Total" rows.

* convert Var1 (which holds subyr) from string to numeric and rename.

* 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 SameChild).
```

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

```
execute.
```

* If > 5% don't match on DOB and sex, flag as a failed.

```
if SameChild > 5 DQFailedCheck_SameChild = 1.
if sysmis(DQFailedCheck_SameChild) DQFailedCheck_SameChild=0.
sort cases by staterr.
execute.
save outfile = 'DQ NCANDS\DQ_SameChild Recur ' + YYstr + YY2str + '.sav'
  /keep staterr SameChild DQFailedCheck_SameChild.
dataset activate SourceData.
dataset close DQ.
```

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

* sort cases BY staterr.

```
match files
  /FILE=*
  /TABLE='DQ NCANDS\DQ_SameChild Recur ' + YYstr + YY2str + '.sav'
  /BY staterr
  /KEEP cfiledat to DQFailedCheck_IDMatch DQFailedCheck_SameChild.
execute.
```

```
dataset close SourceData_Aggregated.
```

```
*****
```

* REMOVE STATES THAT EXCEED DQ LIMITS ON ANY OF CHECKS

```
*****
```

* Review and verify states that will be excluded.

```
*****
```

temporary.

```
select if DQFailedCheck_Age_EitherYear = 1 OR DQFailedCheck_IDMatch = 1 OR DQFailedCheck_SameChild = 1.
```

```
ctables
  /VLABELS VARIABLES=staterr DISPLAY=NONE /VLABELS VARIABLES=DQFailedCheck_Age_EitherYear DQFailedCheck_IDMatch DQFailedCheck_SameChild
  DISPLAY=LABEL
  /TABLE staterr [C][COUNT F40.0] BY DQFailedCheck_Age_EitherYear [C] + DQFailedCheck_IDMatch [C] + DQFailedCheck_SameChild [C]
  /SLABELS VISIBLE=NO
  /CATEGORIES VARIABLES=staterr ORDER=A KEY=VALUE EMPTY=EXCLUDE
  /CATEGORIES VARIABLES=DQFailedCheck_Age_EitherYear [1] EMPTY=INCLUDE
  /CATEGORIES VARIABLES=DQFailedCheck_IDMatch [1] EMPTY=INCLUDE
  /CATEGORIES VARIABLES=DQFailedCheck_SameChild [1] EMPTY=INCLUDE.
```

* Select only states that met the DQ checks.

```
select if DQFailedCheck_Age_EitherYear = 0 AND DQFailedCheck_IDMatch = 0 AND DQFailedCheck_SameChild = 0.
```

execute.

```
delete variables DQFailedCheck_Age DQFailedCheck_Age_EitherYear DQFailedCheck_IDMatch DQFailedCheck_SameChild.
```

* Count the number of states remaining.

```
compute numstates = 1.
if (staterr eq lag(staterr))numstates = 0.
frequencies numstates.
```

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

```
* Select only records with report dates on or after the beginning of the first fiscal year.
select if rptdt ge date.dmy (01,10,YYYY-1).
execute.
```

```
* Exclude children over 17 years; set unborn to 0 (unborn are grouped into the 0-3 months age group).
recode chage (77 = 0).
frequencies chage.
select if chage le 17.
frequencies chage.
```

```
*****
* DERIVED VARIABLES AND PROCESSING
*****
```

```
* Determine the time between reports for 14 day rollup.
sort cases by staterr chid rptdt.
do if (staterr = lag(staterr) and chid = lag(chid)).
compute timebetweenrpts = datediff ( rptdt,lag(rptdt), 'days').
end if.
* frequencies timebetweenrpts.
```

```
* Apply 14 day roll up.
recode timebetweenrpts (sysmis = 999).
* frequencies timebetweenrpts.
select if timebetweenrpts gt 14.
* frequencies timebetweenrpts.
```

```
* Count the relative report position for a victim.
compute posit = 1.
if (staterr eq lag(staterr) and chid eq lag(chid))posit = lag(posit)+1.
frequencies posit.
crosstabs posit by maldeath.
```

```
* Exclude fatalities if they happened in the first report.
compute keep = 1.
if (posit eq 1 and maldeath eq 1)keep = 0.
frequencies keep.
```

select if keep = 1.
frequencies keep maldeath.

* Select only the first two reports for a victim. Recurrence is only computed using the first two reports, so
* subsequent reports are of no interest.

select if (posit le 2).
frequencies posit.

* Aggregate to create a unique file.
dataset declare SourceData_Aggregated.
aggregate outfile = 'SourceData_Aggregated'
/break = staterr chid
/TwelveMoCohort = first(TwelveMoCohort)
/FFY = first(subyr)
/county = first(rptcnty)
/dob = first (chbdate)
/ageinyears = first (chage)
/initialrpt = first(rptdt)
/lastrpt = last(rptdt)
/initialincdt = first(inciddt)
/lastincdt = last(inciddt).
dataset activate SourceData_Aggregated.
dataset close SourceData.

* Compute age in months at the time of the initial report.

compute ageinmonths = datediff(initialrpt,dob, 'months').
* frequencies ageinmonths.

* Some children have a negative ageinmonths due to dob > initialrpt. These are plausible and involve allegations
* concerning children in utero. Recode to 0 for 0-3 months age group, which includes unborn.

recode ageinmonths (lowest thru -1 = 0).
variable labels ageinmonths 'Age in months at initial report'.
value labels ageinmonths 0 'less than 1 month includes unborn'.
* frequencies ageinmonths.

recode ageinyears (lowest thru -1 = 0).
variable labels ageinyears 'Age in years at initial report'.
value labels ageinyears 0 'Less than 1 year includes unborn' 99 'Unknown age'.
frequencies ageinyears.

* Flag cases where recurrence occurs.

* Compute the time between the first and the last report.
compute daysbetweenrpts = datediff(lastrpt,initialrpt, 'days').
* frequencies daysbetweenrpts.

* If the time between reports is greater than 0 and less than or equal to 365 days, recurrence occurs.
compute recur = 0.
if (daysbetweenrpts gt 0 and daysbetweenrpts le 365) recur = 1.
frequencies recur.

* If the initial incident date is the same as the last incident date, set recur to 0.
do if (initialincdt ge 1 and lastincdt ge 1).
if (initialincdt eq lastincdt)recur = 0.
end if.
frequencies recur.
variable labels recur 'Maltx recurrence numerator - Child had a recurrence of maltx within 12 months of his initial report'.
value labels recur
 0 'No recurrence occurred'
 1 'Yes recurrence occurred'.
formats recur (F1.0).
frequencies recur.

* Select only records where the first report date occurred in the first fiscal year (this uses two submission periods).
select if (initialrpt le date.mdy(09,30,YYYY)).
frequencies recur.

* Rename and create variables consistent with other indicators.

* Add state full names and abbreviations by merging these in from states.sav data file. These are useful for
* reporting purposes, custom tables, etc.
get file 'Fixed files\states.sav'.
dataset name states window=front.
sort cases by staterr.
save outfile='Fixed files\states.sav'.
dataset activate SourceData_Aggregated.
dataset close states.
* sort cases by staterr.
match files /FILE=*
 /TABLE='Fixed files\states.sav'
 /BY staterr.
execute.

* Rename.
rename variables (chid ageinmonths ageinyears recur = ChildID AgeMos AgeYrs Num_Child).

* CALCULATE AGE AT INITIAL REPORT

* Age in months and age in years at initial report already exist. This syntax recodes them to match the categories and levels used in reporting and risk adjustment.

* Age at initial victimization (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.

* Recode age in months to desired categories.

recode AgeMos (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 AgeMosyrsCat.

* Age at initial victimization (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.

* Children < 1 currently have a value of 0 for AgeYrs. 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 AgeYrs then shift all subsequent ages, so

* age1 = 2, age2 = 3, etc.

compute AgeMosyrs = AgeYrs.

* Recode 1 - 21 to 2 - 22.

recode AgeYrs (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 AgeMosyrs.

* If child is 0-3, make AgeMosyrs = 0.

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

if AgeMosyrsCat = 0 AgeMosyrs = 0.

if AgeMosyrsCat = 1 AgeMosyrs = 1.

execute.

variable labels

AgeMos 'Age at initial report in months'

AgeYrs 'Age at initial report in years (Under 1, 1, 2, 3 ... 21)'

AgeMosyrs 'Age at initial report in months (0-3 mo, 4-11 mo) and years (1, 2, 3 ... 21)'

AgeMosyrsCat 'Age at initial report 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'
!ENDDFINE.

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.

value labels

AgeMos
8888 'Invalid (age is negative or > 21)'
9999 'Cannot calculate (DOB missing)'
/AgeYrs
!AgeYrs
8888 'Invalid (age is negative or > 21)'
9999 'Cannot calculate (DOB missing)'
/AgeMosyrs
!AgeMosYrs
8888 'Invalid (age is negative or > 21)'
9999 'Cannot calculate (DOB missing)'
/AgeMosyrsCat
!AgeMosYrsCat
8888 'Invalid (age is negative or > 21)'
9999 'Cannot calculate (DOB missing)'.

formats AgeYrs AgeMos AgeMosyrsCat AgeMosyrs (F2.0).

* SUMMARY STATISTICS

* Display observed performance by state and for nation.

ctables

/VLABELS VARIABLES=stateabb DISPLAY=NAME /VLABELS VARIABLES=Num_Child DISPLAY=BOTH
/TABLE stateabb [C][COUNT F40.0, ROWPCT.COUNT PCT40.1] BY Num_Child [C]
/CATEGORIES VARIABLES=stateabb ORDER=A KEY=VALUE EMPTY=EXCLUDE TOTAL=YES POSITION=AFTER
/CATEGORIES VARIABLES=Num_Child ORDER=A KEY=VALUE EMPTY=EXCLUDE.

* Create variables holding state and national performance.

* State-level data.

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

aggregate

```
/OUTFILE=* MODE=ADDVARIABLES
/PRESORTED
/BREAK=state
/Num_State=SUM(Num_Child)
/Den_State=N.
```

```
compute Perf_State = Num_State / Den_State.
compute Perf_State_MP = (Num_State / Den_State) * 100.
execute.
```

* National-level data.

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

```
compute Perf_Nation = Num_Nation / Den_Nation.
compute Perf_Nation_MP = (Num_Nation / Den_Nation) * 100.
execute.
```

```
formats Perf_State Perf_Nation (F8.5).
```

variable labels

Den_State 'Maltx recurrence denominator - Number of children in state who were victimized in the first 12-month period'

Num_State 'Maltx recurrence numerator - Among children in the denominator, number of children in state who had a recurrence of maltx within 12 months of their initial victimization'

Perf_State 'Maltx recurrence - Percentage of children in state who had a recurrence of maltx within 12 months of their initial victimization'

Den_Nation 'Maltx recurrence denominator - Number of children in nation who were victimized in the first 12-month period'

Num_Nation 'Maltx recurrence numerator - Among children in the denominator, number of children in nation who had a recurrence of maltx within 12 months of their initial victimization'

Perf_Nation 'Maltx recurrence - Percentage of children in nation who had a recurrence of maltx within 12 months of their initial victimization'.

```
*****
```

* OUTPUT FILES

```
*****
```

* Save.

```
save outfile='Performance observed child\CFSR 3 - Observed perf for maltx recurrence ' + YYstr + YY2str + '.sav'
/compressed.
```

* Save file for STATA (for multi-level modeling).

```
rename variables (AgeMosyrs = ChildAge).
```

```
save translate OUTFILE='Performance observed child\CFSR 3 - Observed perf for maltx recurrence ' + YYstr + YY2str + '.dta'
```

```
/TYPE=STATA
```

```
/VERSION=8
```

```
/EDITION=SE
```

```
/MAP
```

```
/REPLACE
/KEEP=state stateabb TwelveMoCohort ChildID Num_Child ChildAge Den_State
Num_State Perf_State Den_Nation Num_Nation Perf_Nation.

* Change name back.
rename variables (ChildAge = AgeMosyrs).

* 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 recurrence".
execute.

save outfile='Performance observed state\CFSR 3 - Observed perf for maltx recurrence State file ' + YYstr + YY2str + '.sav'
/keep state stateabb statetxt Indicator TwelveMoCohort Perf_State.

dataset activate SourceData_Aggregated.
dataset close OneRecordPerState.

output save OUTFILE='Output\CFSR 3 - Observed perf for maltx recurrence ' + YYstr + YY2str + '.spv'.

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

* No need to save anything.
new file.
dataset name AllDone WINDOW=FRONT.
dataset close SourceData_Aggregated.
```