

set printback=on.

\*\*\*\*\*  
\* CFSR 3 - CREATE AFCARS SOURCE DATA  
\*\*\*\*\*

\* 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 6-month AFCARS .sav files from all states in the 'CFSR3\Submissions AFCARS\SixMonthFiles' folder. See the next section for guidance regarding which 6-month submissions to include.

set printback=on.

\*\*\*\*\*  
\* MERGE 6-MONTH SUBMISSIONS  
\*\*\*\*\*

\* Calculating performance on CFSR 3 indicators requires multiple 6-month submissions. The number of submissions needed depends on the indicator. The date of those submissions (i.e., the reporting period) depends on the 12-month cohort for which performance will be measured. For example, to measure performance for Permanency in 12 months for children entering foster care (and the companion Re-entry indicator) requires six, 6-month submissions (i.e., three years of data). So, to measure Permanency for children who entered during 11B12A requires these 6-month submissions: 11B, 12A, 12B, 13A, 13B, and 14A.

\* This step will merge together all 6-month submissions stored in the 'SixMonthFiles' folder. In determining what 6-month files to include, CB recommends starting with 09B and ending with the most recent submission available (i.e., from the latest reporting period). In other words, CB recommends building and maintaining a historical dataset starting with 09B (although there is no problem with adding earlier periods).

\* Merge together all files in the submissions folder.  
\*\*\*\*\*

\* Set working directory.  
cd 'C:\CFSR3'.  
show directory.

\* Close any existing data files or datasets.  
dataset close all.  
new file.

\* Merge files. This syntax requires that SPSS Python Essentials be installed. It is installed by default beginning with SPSS Version 22.

```
begin program.
rdir = 'C:/CF3R3/Submissions AFCARS/SixMonthFiles' #Please specify folder containing .sav files.
import spss,os
fils = sorted([fil for fil in os.listdir(rdir) if fil.endswith('.sav')])
spss.Submit('get file "%s/%s".'%(rdir,fils.pop(0)))
for rep in range(len(fils)/49 + 1):
    spss.Submit('add files file=*%s.%s'.join(['file="%s"'%os.path.join(rdir,fil) for fil in fils[49*rep:49*rep + min(49,len(fils)-49*rep)]]))
spss.Submit('exe.')
end program.
```

dataset name SourceFile.

```
frequencies state.
crosstabs
  /TABLES=repdatyr BY repdatmo
  /FORMAT=AVALUE TABLES
  /CELLS=COUNT
  /COUNT ROUND CELL.
```

```
save outfile='Submissions AFCARS/Merged AFCARS submissions.sav'
  /COMPRESSED.
```

```
*****
* FILE PREP
*****
```

```
* Create a unique, numeric ID (Child ID) across the entire file.
*****
```

```
* Create a unique ID (recnumbr2) across the entire file (state FIPS + recnumber).
sort cases by state recnumbr.
string statestring (A2).
compute statestring = STRING(state, F2.0).
execute.
string recnumbr2 (A50).
compute recnumbr2=concat(statestring,recnumbr).
execute.
```

```
* Create a unique, numeric version of recnumbr2, called ChildID. ChildID will be used later for sorting, breaking,
* and other functions. It is more efficient and reliable than using the recnumbr2, which is a string and, for some
* states, contains highly specialized characters.
```

```
autorecode VARIABLES=recnumbr2
  /INTO ChildID.
value labels ChildID.
delete variables statestring recnumbr2.
```

\* Add state names and abbreviations.

\*\*\*\*\*

\* 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 state.

save outfile='Fixed files\states.sav'.

dataset activate SourceFile.

dataset close states.

match files /FILE=\*

  /TABLE='Fixed files\states.sav'

  /BY state.

execute.

\* Create date fields.

\*\*\*\*\*

\* Start and end dates for each 6-month reporting period in the merged file.

\* ... A files (Oct 1 - Mar 31).

if repdatmo = 3 DtReportBeg = date.mdy(10,1,repdatyr-1).

if repdatmo = 3 DtReportEnd= date.mdy(3,31,repdatyr).

\* ... B files (Apr 1 - Sep 30).

if repdatmo = 9 DtReportBeg = date.mdy(4,1,repdatyr).

if repdatmo = 9 DtReportEnd = date.mdy(9,30,repdatyr).

\* Date of the last 6-month reporting period in the merged file.

aggregate

  /OUTFILE=\* MODE=ADDVARIABLES

  /BREAK=

  /DtReportEndFinal=MAX(DtReportEnd).

\* Other key dates. Some of these are not used in the CFSR syntax, and are created for convenience in case

\* users are interested in making use of them).

compute DtReview=date.mdy(pedrevmo, pedrevda,pedrevyr).

compute DtBirth=date.mdy(dobmo,dobda,dobyr).

compute DtFirRem=date.mdy(rem1mo,rem1da,rem1yr).

compute DtPriorDisch =date.mdy(dlstfcmo,dlstfcda,dlstfcyr).

compute DtLatRem=date.mdy(latremmo,latremda,latremyr).

compute DtLatRemTrans=date.mdy(remtrnmo, remtrnda, remtrnyr).

compute DtCurSet=date.mdy(cursetmo,cursetda,cursetyr).

compute DtTPRMom=date.mdy(prtmommo, prtmomda, prtmomyr).

compute DtTPRDad=date.mdy(prtdadmo, prtdadda, prtdadyr).

```
compute DtDisch=date.mdy(dodfcmo,dodfcda,dodfcyr).
compute DtDischTrans=date.mdy(dodtrnmo,dodtrnda,dodtrnyr).
execute.
```

\* Formatting.

variable labels

```
DtReportBeg 'Start date of the 6-month reporting period (10/1/YYYY or 4/1/YYYY)'
DtReportEnd 'End date of the 6-month reporting period (3/31/YYYY or 9/30/YYYY)'
DtReportEndFinal 'Date of the last 6-month reporting period in the file'
DtReview 'Date of most recent periodic review'
DtBirth 'Date of birth'
DtFirRem 'Date of first removal from home'
DtPriorDisch 'Date of discharge from last foster care episode'
DtLatRem 'Date of latest removal from home'
DtLatRemTrans 'Date (transaction) of latest removal from home'
DtCurSet 'Date of placement in current foster care setting'
DtTPRMom 'Date of mother parental rights termination'
DtTPRDad 'Date of father parental rights termination'
DtDisch 'Date of discharge from foster care'
DtDischTrans 'Date (transaction) of discharge from foster care'.
formats DtReportBeg to DtDischTrans (adate10).
```

\* Delete original date fields.

delete variables

```
repatyr repdatmo
pedevyr pedrevmo pedrevda
dobyр dobmo dobda
rem1yr rem1mo rem1da
dlstfcyr dlstfcmo dlstfcda
latremyr latremmo latremda
remtrnyr remtrnmo remtrnda
cursetyr cursetmo cursetda
prtmomyr prtmommo prtmomda
prtdadyr prtdadmo prtdadda
dodfcyr dodfcmo dodfcda
dodtrnyr dodtrnmo dodtrnda.
```

\* Delete duplicate records.

```
*****
```

\* Identify children reported two or more times during the same 6-month reporting period. These occur rarely, as

\* states are advised to report only the most recent episode per 6-month reporting period.

```
SORT CASES BY ChildID(A) DtReportBeg(A) DtLatRem(A) DtDisch(A).
```

```
MATCH FILES
```

```
/FILE=*
```

```

/BY ChildID DtReportBeg
/FIRST=PrimaryFirst
/LAST=PrimaryLast.
DO IF (PrimaryFirst).
COMPUTE MatchSequence=1-PrimaryLast.
ELSE.
COMPUTE MatchSequence=MatchSequence+1.
END IF.
LEAVE MatchSequence.
FORMATS MatchSequence (f7).
COMPUTE InDupGrp=MatchSequence>0.
SORT CASES InDupGrp(D).
MATCH FILES
/FILE=*
/DROP=PrimaryFirst MatchSequence.
VARIABLE LABELS PrimaryLast 'Indicator of each last matching case as Primary'.
VALUE LABELS PrimaryLast 0 'Duplicate Case' 1 'Primary Case'.
VARIABLE LEVEL PrimaryLast (ORDINAL).
FREQUENCIES VARIABLES=PrimaryLast.
EXECUTE.

```

\* Document states that reported the same children two or more times in a report period, and the number of  
\* duplicates reported.

temporary.

select if InDupGrp = 1 and PrimaryLast = 0.

crosstabs

```

/TABLES=state BY PrimaryLast
/FORMAT=AVALUE TABLES
/CELLS=COUNT
/COUNT ROUND CELL.

```

\* Discard duplicates. The sort order means we try to keep the record with the latest removal date (if it differs)

\* and, among those, the record with a discharge date (if present).

select if PrimaryLast = 1.

execute.

delete variables PrimaryLast InDupGrp.

\*\*\*\*\*

\* DATA QUALITY - AFCARS CROSS-FILE CHECKS

\*\*\*\*\*

\* Cross-file checks look across every two consecutive 6-month reporting periods in the merged file to determine

\* if a child was reported in the next 6-month report period. Two checks are then carried out:

- \* 1. Dropped records
- \* 2. AFCARS IDs don't match from one period to the next

\* These checks are not performed on records that occur in the last reporting period in the merged file, because  
\* there is no subsequent 6-month reporting period to look for.

\* Preparation - Calculate TimeBetweenReports.

\*\*\*\*\*

\* First, calculate the time in months between the child's record and any subsequent report that may exist for  
\* the child. The two DQ checks will use this information.  
sort cases BY ChildID(A) DtReportBeg(A).

\* For each record, make a copy of DtReportBeg, which stores the start date of the 6-month reporting period.  
compute DtReportBeg1 = DtReportBeg.  
execute.  
formats DtReportBeg1 (adate10).

\* Create a variable called Next6MoReport. If the child was reported in a subsequent 6-month period (i.e.,  
\* another DtReportBeg for this child is found), copy the start date of that subsequent 6-month submission  
\* into Next6MoReport. If the child was not reported in a subsequent submission, leave Next6MoReport blank.  
\* Note: For a new child, DtReportBeg1 is set to missing; otherwise, the start date will get assigned to the  
\* previous child.  
set workspace 200000.  
if (ChildID<>lag(ChildID)) DtReportBeg1 = \$systemis.  
create Next6MoReport=Lead(DtReportBeg1,1).  
execute.  
formats Next6MoReport (adate10).  
set workspace 6200.

\* Calculate the number of months between the start date of the current 6-month reporting period and the start  
\* date of the subsequent 6-month report period that was found for this child. If the reporting period is the last  
\* reporting period in the merged file, then there is no subsequent report to look for, so code that as 888 (Not  
\* Applicable).  
do if DtReportEnd eq DtReportEndFinal.  
compute TimeBetweenReports = 888.  
else.  
compute TimeBetweenReports = datediff(Next6MoReport,DtReportBeg,"months").  
end if.  
execute.  
freq TimeBetweenReports.

\* Possible results for TimeBetweenReports:

\* ... blank = a subsequent report was not found for this child

\* ... 6 = a subsequent report for this child was found, and it occurred in the next 6-month reporting period

\* ... 12 or more = a subsequent report for this child was found, but it occurred in a later reporting period (i.e., not the immediate next one).  
\* ... 888 = not applicable (report is in the last reporting period in the merged file).

\* Dropped records.

\*\*\*\*\*

\* It is a dropped record if the 6-month record is missing a DtDisch and a subsequent report was not found for this child (TimeBetweenReports = BLANK) or a subsequent report was found for this child but it occurred in a reporting period other than the immediate next one (TimeBetweenReports >= 12 months).  
do if sysmis(DtDisch).  
if sysmis(TimeBetweenReports) OR (TimeBetweenReports ge 12 AND TimeBetweenReports lt 888) DQ\_Dropped = 1.  
end if.  
execute.

\* AFCARS IDs don't match from one period to next.

\*\*\*\*\*

\* It is a non-match if the child was not reported in the next 6-month period (TimeBetweenReports = BLANK) or a subsequent report was found for this child but it occurred in a reporting period other than the immediate next one (TimeBetweenReports >= 12 months). These records are not in error and not a problem unless the percentage of non-matches for the entire state exceeds the DQ limit.  
if sysmis(TimeBetweenReports) OR (TimeBetweenReports ge 12 AND TimeBetweenReports lt 888) DQ\_IDNoMatchNext6Mo = 1.  
execute.

delete variables DtReportBeg1 Next6MoReport TimeBetweenReports.

\*\*\*\*\*

#### \* DATA QUALITY - AFCARS WITHIN-FILE CHECKS

\*\*\*\*\*

\* These checks are run on each 6-month record in the combined file, within each 6-month report period.  
\* #variables are scratch variables - they are created temporarily and then automatically removed.

\* Missing date of birth.  
\* Missing date of latest removal.  
\* Missing number of placement settings.  
if sysmis(DtBirth) DQ\_missDOB = 1.  
if sysmis(DtLatRem) DQ\_missDtLatRem = 1.  
if sysmis(numplep) DQ\_missNumPlep = 1.  
execute.

\* Date of birth is after the date of latest removal (produces a negative age of entry).  
if DtBirth > DtLatRem DQ\_DOBgtDtLatRem = 1.

\* Date of birth is after date of discharge (produces a negative age at exit).

if DtBirth > DtDisch DQ\_DOBgtDtDisch = 1.

\* Time between date of birth and date of latest removal is > 21 yrs (produces age at entry > 21 yrs).

compute #AgeN = datediff(DtLatRem,DtBirth,"years").

if #AgeN > 21 DQ\_gt21DOBtoDtLatRem = 1.

\* Time between date of birth and date of discharge is > 21 yrs (produces age at exit > 21 yrs).

compute #AgeX = datediff(DtDisch,DtBirth,"years").

if #AgeX > 21 DQ\_gt21DOBtoDtDisch = 1.

\* Time between date of latest removal and date of discharge is > 21 yrs (produces a LOS > 21 yrs).

compute #LOSN = datediff(DtDisch, DtLatRem, "days").

if #LOSN > 7665 DQ\_gt21DtDischtoDtLatRem = 1.

\* Date of latest removal is equal to the date of discharge (produces a LOS of 0 to 24 hrs).

if DtDisch = DtLatRem DQ\_DtDischeqDtLatRem = 1.

\* Date of discharge is before the date of latest removal (produces a negative LOS).

if DtDisch < DtLatRem DQ\_DtDischlDtLatRem = 1.

\* Date of discharge exists but the reason for discharge is missing.

if not sysmis(DtDisch) and sysmis(disreasn) DQ\_missDisreasn = 1.

\* Child is on first removal (total # of removals from home to date = 1).

\* Not a problem unless the % of children on their first removal exceeds the DQ limit.

if totalrem = 1 DQ\_totalrem1 = 1.

execute.

\* Formatting

\*\*\*\*\*

\* Add 0s in place of missing so they show up in custom tables.

recode DQ\_Dropped to DQ\_totalrem1 (SYSMIS=0).

execute.

\* As noted earlier, records with a 1 for DQ\_IDNoMatchNext6Mo are not in error and not a problem

\* unless the percentage of non-matches for the entire state exceeds the DQ limit.

value labels DQ\_Dropped to DQ\_totalrem1

0 'Okay'

1 'Potential problem'.

formats DQ\_Dropped to DQ\_totalrem1 (F2.0).

variable labels

DQ\_Dropped 'Record is missing a date of discharge, suggesting child is still in care, but a record in the next 6-month period does not exist'

DQ\_IDNoMatchNext6Mo 'No match for a given recnumbr in the next 6-month period. Not a problem unless the percentage of non-matches for a given state exceeds the DQ limit'  
DQ\_missDOB 'Date of birth is missing'  
DQ\_missDtLatRem 'Date of latest removal from home is missing'  
DQ\_missNumPlep 'Number of placements in current foster care episode is missing'  
DQ\_DOBgtDtLatRem 'Date of birth is after the date of latest removal from home (produces a negative age of entry)'  
DQ\_DOBgtDtDisch 'Date of birth is after the date of discharge from most recent foster care episode (produces a negative age at exit)'  
DQ\_gt21DOBtoDtLatRem 'Time between date of birth and date of latest removal from home is > 21 yrs (produces age at entry > 21 yrs)'  
DQ\_gt21DOBtoDtDisch 'Time between date of birth and date of discharge from foster care is > 21 yrs (produces age at exit > 21 yrs)'  
DQ\_gt21DtDischtoDtLatRem 'Time between date of latest removal from home and date of discharge from most recent foster care episode is > 21 yrs (produces a LOS > 21 yrs)'  
DQ\_DtDischeqDtLatRem 'Date of latest removal from home is the same day as the date of discharge from most recent foster care episode (produces a LOS of 0 days)'  
DQ\_DtDischlDtLatRem 'Date of discharge from most recent foster care episode is before the date of latest removal from home (produces a negative LOS)'  
DQ\_missDisreasn 'Date of discharge from most recent foster care episode exists but the reason for discharge is missing'  
DQ\_totalrem1 'Child is on first removal (total # of removals from home to date = 1). Not a problem unless the percentage of children on their first removal for a given state exceeds the DQ limit'.

\*\*\*\*\*

\* CALCULATE DERIVED VARIABLES

\*\*\*\*\*

\* These derived variables are calculated for each record in the file. Some will be used in the indicator-specific  
\* syntax; others may be useful for other reporting and analyses.

\* Entry and exit years

\*\*\*\*\*

\* FFY child entered.

compute EntryYr=xdate.year(DtLatRem).

if range(xdate.month(DtLatRem),10,12) EntryYr=EntryYr+1.

\* If date of latest removal is missing, recode to 9999.

if DQ\_missDtLatRem = 1 EntryYr = 9999.

execute.

\* FFY child exited.

compute ExitYr=xdate.year(DtDisch).

if range(xdate.month(DtDisch),10,12) ExitYr=ExitYr+1.

\* If date of discharge is missing, assume child is still in care.

if sysmis(DtDisch) ExitYr = 7777.

execute.

variable labels

EntryYr 'FFY child entered'

ExitYr 'FFY child exited care (or 7777 if child is still in care)'.  
value labels

EntryYr 9999 'Cannot calculate (DtLatRem missing)'

/ ExitYr 7777 'Child still in care (DtDisch missing)'.

formats EntryYr ExitYr (F4.0).

\* Age at entry (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 at entry in months.

compute AgeNmos = datediff(DtLatRem,DtBirth,"months").

\* Record age in months to desired categories.

recode AgeNmos (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 AgeNmosyrsCat.

execute.

\* Age at entry (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 at entry in years.

compute AgeNyrs = datediff(DtLatRem,DtBirth,"years").

\* Children < 1 currently have a value of 0 for AgeNyrs. 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 AgeNyrs then shift all subsequent ages, so age1 = 2, age2 = 3, etc.

compute AgeNmosyrs = AgeNyrs.

\* Recode 1 - 21 to 2 - 22.

recode AgeNyrs (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  
AgeNmosyrs.

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

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

if AgeNmosyrsCat = 0 AgeNmosyrs = 0.

if AgeNmosyrsCat = 1 AgeNmosyrs = 1.

execute.

\* Handle problem values.

\*\*\*\*\*

\* If age at entry is negative or > 21 yrs, recode to 8888.

do if DQ\_DOBgtDtLatRem = 1 or DQ\_gt21DOBtoDtLatRem = 1.

recode AgeNmos AgeNyrs AgeNmosyrsCat AgeNmosyrs (else = 8888).

end if.

\* If age at entry can't be calculated, recode to 9999 (missing DOB or DtLatRem).

do if DQ\_missDOB = 1 or DQ\_missDtLatRem = 1.

recode AgeNmos AgeNyrs AgeNmosyrsCat AgeNmosyrs (else = 9999).

end if.

execute.

\* Age at exit (categorical), specified as 0-3, 4-11, 1-5, 6-10, 11-16, 17, 18-21.

\* Useful for reporting and descriptive statistics.

\*\*\*\*\*

\* Calculate age at exit in months (with 777 for still in care).

compute AgeXmos = datediff(DtDisch,DtBirth,"months").

if sysmis(DtDisch) AgeXmos = 7777.

\* Recode age in months to desired categories.

recode AgeXmos (7777=7777) (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 AgeXmosyrsCat.

execute.

\* Age at exit (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 at exit in years (with 777 for still in care)..

compute AgeXyrs = datediff(DtDisch,DtBirth,"years").

if sysmis(DtDisch) AgeXyrs = 7777.

execute.

\* Children < 1 currently have a value of 0. 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 AgXNyrs then shift all subsequent ages, so age1 = 2, age2 = 3, etc.

compute AgeXmosyrs = AgeXyrs.

\* Recode 1 - 21 to 2 - 22.

recode AgeXyrs (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  
AgeXmosyrs.

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

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

if AgeXmosyrsCat = 0 AgeXmosyrs = 0.

if AgeXmosyrsCat = 1 AgeXmosyrs = 1.

execute.

\* Handle problem values.

\*\*\*\*\*

\* If age at exit is negative or > 21 yrs, recode to 8888.

do if DQ\_DOBgtDtDisch = 1 or DQ\_gt21DOBtoDtDisch = 1.

recode AgeXmos AgeXyrs AgeXmosyrsCat AgeXmosyrs (else = 8888).

end if.

\* If age at exit can't be calculated, recode to 9999 (missing DOB).

do if DQ\_missDOB = 1.

recode AgeXmos AgeXyrs AgeXmosyrsCat AgeXmosyrs (else = 9999).

end if.

execute.

\* Formatting.

\*\*\*\*\*

variable labels

EntryYr 'FFY child entered'

AgeNmos 'Age at entry in months'

AgeNyrs 'Age at entry in years (Under 1, 1, 2, 3 ... 21)'

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

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

AgeXmos 'Age at exit in months'

AgeXyrs 'Age at exit in years (Under 1, 1, 2, 3 ... 21, 7777 for still in care)'

AgeXmosyrs 'Age at exit in months (0-3 mo, 4-11 mo) and years (1, 2, 3 ... 21, 7777 for still in care)'

AgeXmosyrsCat 'Age at exit in months (0-3 mo, 4-11 mo) and years in categories (1-5, 6-10, ... 7777 for still in care)'.

\* 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'

!ENDDEFINE.

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'  
!ENDDDEFINE.

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'  
!ENDDDEFINE.

\* Age at entry.

value labels

AgeNmos

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

9999 'Cannot calculate (DOB or date of latest removal missing)'

/AgeNyrs

!AgeYrs

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

9999 'Cannot calculate (DOB or date of latest removal missing)'

/AgeNmosyrs

!AgeMosYrs

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

9999 'Cannot calculate (DOB or date of latest removal missing)'

/AgeNmosyrsCat

!AgeMosYrsCat

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

9999 'Cannot calculate (DOB or date of latest removal missing)'.

\* Age at exit.

value labels

AgeXmos

7777 'Not applicable (child still in care as of the end of the 6-month report)'

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

9999 'Cannot calculate (DOB or date of latest removal missing)'

/AgeXyrs

!AgeYrs

7777 'Not applicable (child still in care as of the end of the 6-month report)'

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

9999 'Cannot calculate (DOB missing)'

/AgeXmosyrs

!AgeMosYrs

7777 'Not applicable (child still in care as of the end of the 6-month report)'

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

9999 'Cannot calculate (DOB or date of latest removal missing)'

/ AgeXmosyrsCat

!AgeMosYrsCat

7777 'Not applicable (child still in care as of the end of the 6-month report)'

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

9999 'Cannot calculate (DOB missing)'.

formats AgeNmos to AgeXmosyrs (F2.0).

\* DisReason1

\*\*\*\*\*

\* Same as the original reason reported by the state, except when disreasn is missing, uses 7777 for still in care.

\* Note: The number of reports where DisReasn1 = 7777 will not equal the number of reports where AgeX = 7777,

\* because for DisReasn1 we define still in care as sysmis(disreasn) or disreasn = 0, and for AgeX we define

\* still in care as sysmis(DtDisch).

compute DisReason1 = disreasn.

if sysmis(disreasn) or disreasn = 0 DisReason1 = 7777.

execute.

variable labels DisReason1 'Discharge reason (or 7777 if child still in care)'.

value labels DisReason1

1 'Reunification'

2 'Living with a Relative'

3 'Adoption'

4 'Emancipation'

5 'Guardianship'  
6 'Transfer to another agency'  
7 'Runaway'  
8 'Death of child'  
7777 'Child was still in care as of the end of the 6-month report'.  
formats DisReason1 (F4.0).

\* DisReason2  
\*\*\*\*\*

\* Groups disreasns into four categories: Perm, NonPerm, Death, or still in care.  
recode DisReason1 (1,2,3,5=1) (4,6,7=2) (8=3) (7777=7777) into DisReason2.  
execute.

variable labels DisReason2 'Discharge reason (perm, non perm, death) or still in care (7777)'.  
value labels DisReason2  
1 'Permanency (all four types)'  
2 'Non-Permanency'  
3 'Child died'  
7777 'Child was still in care as of the end of the 6-month report'.  
formats DisReason2 (F4.0).

\* Total number of removals (categorical)  
\*\*\*\*\*

\* For not used for CFSR 3 performance, but may be useful for reporting and descriptive statistics.

recode totalrem (1=1) (2=2) (3=3) (4 thru Highest=4) (else=sysmis) into TRemCat.  
execute.

variable labels TRemCat 'Total number of removals from home (categorical)'.  
value labels TRemCat  
1 '1'  
2 '2'  
3 '3'  
4 '4 or more'.  
formats TRemCat (F2.0).

\* Save.  
\*\*\*\*\*

save outfile='CFSR 3 AFCARS source data.sav'  
/COMPRESSED.

\* Save output.  
output save OUTFILE='Output\CFSR 3 - Create AFCARS source data.spv'.