# Session 10: Batch Edit and Export

At the end of this session participants will be able to:

- Implement consistency checks in a batch edit application
- Use batch edit to add calculated variables and recoded variables
- Use batch edit to convert checkbox values to yes/no variables for analysis
- Use the export data tool to take data from CSPro to other software packages

## Creating a Batch Edit Application

A batch edit application is like a data entry application but without the forms. It is meant to be run after data entry to detect and fix problems in the data file. A batch edit application takes an input data file and runs logic on it. It generates both a report, called a listing file, and optionally an output data file which is a modified version of the input file. A batch edit application never changes the input file.

To create a batch edit application you choose File→New from CSPro and choose Batch Edit Application. You then choose a dictionary. This is usually the same dictionary that you used for data entry.

The user interface for working with batch applications is similar to the one for working with data entry applications except that there are no forms. Instead there is a tab for edits. Just like in data entry, you add logic to PROCS. Instead of running interactively, all the error messages are written out to a log file for review after the whole program has run.

## Checking for Errors

To add consistency checks we proceed just as we did in our data entry application by adding logic to the appropriate PROC. Let's start with a simple check that the age of first marriage is not greater than the age.

```
// Check for age at first marriage less than current age
if AGE_AT_FIRST_MARRIAGE > AGE then
    errmsg("Age at first marriage greater than age");
endif;
```

Next we run the application but first we need a test data file. You can use the file Popstan2020Raw.csdb. Run the application against this test data. After the application runs, we see the log file. The log file reports that we have a case where this error exists.

```
Process Messages

*** Case [3691141112] has 1 messages (0 E / 0 W / 1U)
    U   -20 Age at first marriage greater than age

User unnumbered messages:

   Line    Freq  Pct.  Message text                                        Denom
   ----    ----  ----  ------------                                       -----
     20       1   0.2  Age at first marriage greater than age               463

CSPRO Executor Normal End
```

To figure out what the problem is we can open up the problem case in data entry. The printout in the listing file contains the case identifiers which we can use to find the case. You can copy the case id from the listing file and use it with Find Case on the Edit menu in CSEntry.

## Correcting Errors

In addition to using batch edit to find errors you can also use it to correct problems by modifying variables in your logic. Let's simply cap the age at first marriage to never be greater than the age.

```
// Check for age at first marriage less than current age
if AGE_AT_FIRST_MARRIAGE > AGE then
    errmsg("Age at first marriage greater than age. Capping age at first
            marriage at age.");
    AGE_AT_FIRST_MARRIAGE = AGE;
endif;
```

When we run this time we will specify an output file: Popstan2020Edited.csdat. The changes we make will only be made to the output file. We can then rerun the batch application on the output file and make sure that you don't have any error messages.

Instead of just assigning the value of `AGE_FIRST_MARRIAGE` we can use the *impute* command which does the assignment just like "=" but also generates a nice report showing the values that were imputed.

```
// Check for age at first marriage less than current age
if AGE_AT_FIRST_MARRIAGE > AGE then
    errmsg("Age at first marriage greater than age. Capping age at first
marriage at age.");
    impute(AGE_AT_FIRST_MARRIAGE, AGE);
endif;
```

The imputation report will be opened in TextViewer after you run the batch application but to see it you will need to go to the Window menu and choose the file that ends in ".frq.lst".

```
                          IMPUTE FREQUENCIES                          Page    1


_____________________________________________________________________________
       Imputed Item AGE_AT_FISRT_MARRIAGE: Age at first marriage - all occurrences
                                  _____________________________ _____________
 Categories                             Frequency       CumFreq       %  Cum %
_____________________________________ _____________________________ _____________
 40                                               1             1  100.0  100.0
_____________________________________ _____________________________ _____________
 TOTAL                                            1             1  100.0  100.0
```

## Adding Calculated Variables

It is often useful to add additional variables to your data file after data collection that are computed from the collected variables. For example, let's add a yes/no/don't know variable to the individual record that determines if the individual is an orphan. First we add the new variable ORPHAN to the dictionary (at the end so that we do not mess up our existing data). Then we add logic to the PROC of our new variable to impute the value. A child is defined as an orphan if both parents are deceased.

```
// Set calculated variable orphan based on mother/father alive
if AGE < 18 then
    if MOTHER_ALIVE = 2 and FATHER_ALIVE = 2 then
        // Mother and father deceased - is orphan
        impute(ORPHAN, 1);
    elseif MOTHER_ALIVE = 1 or FATHER_ALIVE = 1 then
        // Mother or father alive - not orphan
        impute(ORPHAN, 2);
    else
        // Not enough info to determine.
        impute(ORPHAN, 9);
    endif;
endif;
```

Run the program and look at the imputation report to see how many orphans are in our data set.

## Converting Checkboxes to Yes/No

The checkbox makes a nice interface but the resulting data is a string that is hard to interpret and deal with in other software. To make it easier to use for export we can convert from the checkbox to a repeating item with yes/no options. Create a new item in the agriculture record named CROPS_PRODUCED_REPEATING with length 1 and 7 occurrences. Add a value set with Yes – 1 and No – 2.

For each item in the checkbox field that is checked we want to set the corresponding item in the repeating field to one and for each entry in the checkbox field that is not checked we set the corresponding item to two. To find out if an item is checked we use the function ***pos()*** which finds the position of one string in another and returns zero if the string is not found:

```
PROC CROPS_PRODUCED

// Fill in crops repeating item based on crops checkboxes
if pos("A", CROPS_PRODUCED) > 0 then
    CROPS_PRODUCED_REPEATING(1) = 1;
else
    CROPS_PRODUCED_REPEATING(1) = 2;
endif;
if pos("B", CROPS_PRODUCED) > 0 then
    CROPS_PRODUCED_REPEATING(2) = 1;
else
    CROPS_PRODUCED_REPEATING(2) = 2;
endif;
if pos("C", CROPS_PRODUCED) > 0 then
    CROPS_PRODUCED_REPEATING(3) = 1;
else
    CROPS_PRODUCED_REPEATING(3) = 2;
endif;
```

This works but it is a lot of code when the number of options is long. We can simplify it using a loop and a clever trick.

```
PROC CROPS_PRODUCED

// Fill in crops repeating item based on crops checkboxes
numeric i;
string alphabet = "ABCDEFG"; // used to convert from index to letter

// Loop from 1 (Maize) to 7 (Groundnuts) and for
// each crop type find out it if it was checked by
// seeing if the i'th letter of the alphabet is in the
// checkbox field.
do i = 1 while i <= 7
    if pos(alphabet[i:1], CROPS_PRODUCED) > 0 then
        CROPS_PRODUCED_REPEATING(i) = 1;
    else
        CROPS_PRODUCED_REPEATING(i) = 2;
    endif;
enddo;
```

## The Export Data Tool

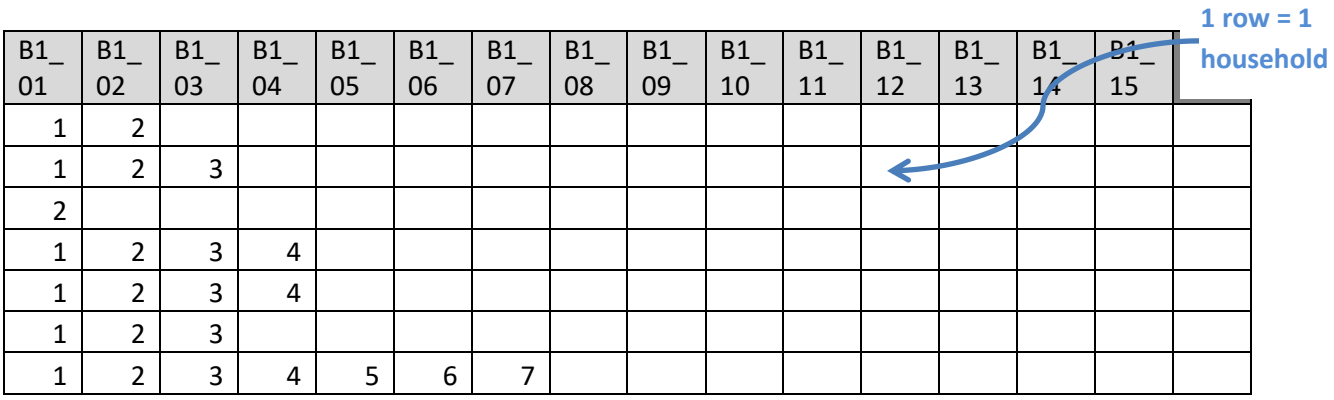
The CSPro export data tool is available from the Tools menu. When you first start Export Data you are prompted to provide a data dictionary. Choose the Popstan2020 dictionary. From the main export screen, you can choose which records/variables to export using the checkboxes next to each one. To start with let's just pick the id items and the first few fields from section F. At the bottom of the export window you can choose the file format to export to. For this exercise we will choose CSV which can be easily opened in Excel. To run the export, click on the traffic light. You are prompted to choose the data file. We will use the Popstan2020Edited.csdb data file. Finally, you are prompted for the name(s) of the exported files. Once the export completes the exported files are displayed in Text Viewer. Let's open

them in Excel and see what we have. Note that each household is saved in a separate line in the Excel file.

| PROVINCE | DISTRICT | ENUMERATION_AREA | AREA TYPE | HOUSEHOLD_NUMBER | F01 | F02 | F03 |
|---|---|---|---|---|---|---|---|
| 2 | 14 | 214 | 1 | 1 | 3 | 1 | 3 |
| 3 | 27 | 301 | 1 | 1 | 9 | 9 | 1 |
| 1 | 1 | 345 | 1 | 5 | 1 | 1 | 1 |
| 3 | 26 | 222 | 2 | 2 | 1 | 4 | 1 |
| 4 | 37 | 422 | 2 | 1 | 1 | 5 | 1 |
| 2 | 2 | 214 | 3 | 1 | 1 | 2 | 4 |
| 1 | 1 | 101 | 1 | 1 | 2 | 5 | 1 |

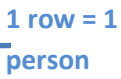
## Exporting Repeating Records

Let's try exporting a few fields from section B: line number, name and sex. Notice for each variable we exported we get 50 columns. Why is that? We are getting one column for each record occurrence. CSPro is still exporting each household on a single row and since there are up to 50 people in the household it generates 50 columns for each variable. Even empty occurrences are still generating columns in the spreadsheet.




| B1_01 | B1_02 | B1_03 | B1_04 | B1_05 | B1_06 | B1_07 | B1_08 | B1_09 | B1_10 | B1_11 | B1_12 | B1_13 | B1_14 | B1_15 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | | | | | | | | | | | | | | |
| 1 | 2 | 3 | | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | | | | | | | | | | | | |
| 1 | 2 | 3 | | | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | | | | | | | | | |

Having a column for each occurrence can complicate working with the data. For example, in this file it is rather tough to do something as simple as count the total number of people. If instead of choosing the default setting of putting multiple record occurrences in a single row (the *All in One Record* setting) try selecting *As Separate Records*. Now we get each person in the household in a separate row. It is important to include the household id items when doing this so that it is clear which people are in which household.




| PROVINCE | DISTRICT | ENUMERATION_AREA | AREA_TYPE | HOUSEHOLD_NUMBER | B1 | B2 | B3 | B5 |
|---|---|---|---|---|---|---|---|---|
| 2 | 14 | 101 | 1 | 1 | 1 | John | 1 | 27 |
| 2 | 14 | 101 | 1 | 1 | 2 | Mary | 2 | 26 |
| 2 | 27 | 200 | 1 | 1 | 1 | Jane | 2 | 40 |
| 2 | 27 | 200 | 1 | 1 | 2 | Billy | 1 | 13 |
| 2 | 27 | 200 | 1 | 1 | 3 | Tina | 2 | 13 |

## Exporting Multiple Record Types

Now let's try exporting the first few items from both the person record (B) and the deaths record (E). With our current settings, Export Data warns us that only items from the first record will be exported. Why? The problem is that if you put each record in its own row then some rows would have household members on them and others would have deaths on them but then the columns would not match.

The solution is to export each record in a separate file by selecting Multiple Files *(one for each record type)* under *Number of Files Created*. Doing this generates two files: PERSON.csv which contains the household members and DEATHS.csv which contains the deaths.

*PERSON.csv*

| PROVINCE | DISTRICT | ENUMERATION_AREA | AREA_TYPE | HOUSEHOLD_NUMBER | B1 | B2 | B3 | B5 |
|---|---|---|---|---|---|---|---|---|
| 2 | 14 | 101 | 1 | 1 | 1 | John | 1 | 27 |
| 2 | 14 | 101 | 1 | 1 | 2 | Mary | 2 | 26 |
| 2 | 27 | 200 | 1 | 1 | 1 | Jane | 2 | 40 |
| 2 | 27 | 200 | 1 | 1 | 2 | Billy | 1 | 13 |
| 2 | 27 | 200 | 1 | 1 | 3 | Tina | 2 | 13 |

*DEATHS.csv*

| PROVINCE | DISTRICT | ENUMERATION_AREA | AREA_TYPE | HOUSEHOLD_NUMBER | E3 | E4 | E5 |
|---|---|---|---|---|---|---|---|
| 2 | 14 | 101 | 1 | 1 | 1 | Erwin | 20140211 |
| 2 | 14 | 101 | 1 | 1 | 2 | Carmine | 20161201 |
| 2 | 27 | 200 | 1 | 1 | 1 | Arnold | 20150113 |

The households in these two files can be linked together based on the id items.

It also possible to have Export Data join together single and multiple records. If you select this option, then one file will be generated for each record with multiple occurrences and the selected columns for all the selected singly occurring records will be added to each row in each of the exported files. For example, if we choose the tenure status (F05) and type of main dwelling (F03) from the single record housing and also choose the first few items from the person record then F05 and F03 will be added to each of the two exported files: PERSON.csv and DEATHS.csv.

*PERSON.CSV*

| PROVINCE | DISTRICT | ENUMERATION_AREA | AREA_TYPE | HOUSEHOLD_NUMBER | F03 | F05 | B1 | B2 |
|---|---|---|---|---|---|---|---|---|
| 2 | 14 | 101 | 1 | 1 | 1 | 1 | 1 | John |
| 2 | 14 | 101 | 1 | 1 | 1 | 1 | 2 | Mary |
| 2 | 27 | 200 | 1 | 1 | 2 | 3 | 1 | Jane |
| 2 | 27 | 200 | 1 | 1 | 2 | 3 | 2 | Billy |
| 2 | 27 | 200 | 1 | 1 | 2 | 3 | 3 | Tina |

Id items

Single record

Multiple record

*DEATHS.CSV*

| PROVINCE | DISTRICT | ENUMERATION_AREA | AREA_TYPE | HOUSEHOLD_NUMBER | F03 | F05 | E3 | E4 |
|---|---|---|---|---|---|---|---|---|
| 2 | 14 | 101 | 1 | 1 | 1 | 1 | 1 | Erwin |
| 2 | 14 | 101 | 1 | 1 | 2 | 1 | 2 | Carmine |
| 2 | 27 | 200 | 1 | 1 | 1 | 3 | 1 | Arnold |

It is not possible to join multiple records to other multiple records from export.

Note that in some cases it may be easier to do the export of the single and multiple records without joining and then do the join in the software that you have imported the data into.

## Importing Data into SAS, SPSS, Stata and R

When exporting data to statistical packages, CSPro generates both a data file and script to run inside the statistical software itself to run the import. For details in how to run this script for each package see the online help for Export Data and look under “How to…”.

## Saving your Export Specification

You can save your export settings as CSPro export specification file (.exf). You can later double click on this file or open it from the Export Data Tool to retrieve all the selections that were made.

## Exercises

1. Modify the batch edit application to add a check for someone with relationship of spouse but marital status that is not married. Print a message for each case found. This should be done in the batch edit application NOT in the data entry application.

2. Modify the batch edit application to add a check that the:

    a. The total number of rooms (F01) is greater than the number of bedrooms (F02)
    b. The type of main dwelling (F03) is consistent with the total numbers of each dwelling type in F04. For example, if the main dwelling is detached house then the total number of detached houses must not be zero.

3. Modify the batch edit application to convert the disabilities (B10) from alpha (used for checkboxes) to a set of numeric yes/no variables. Create the new variables in the dictionary and write logic to set the value of each of the new variables based on the letters.

4. Using the test data, export the household members along with the ID items to the package of your choice (Excel, SPSS, Stata, SAS or R). Use that package to determine the total number of people by sex (total, males, females) and also the number of people by sex for district 01.

5. Export the deaths record and the person record along with the id items into the package of your choice. You should export all the records at once, not one by one. How many households have more than one death? How many households have children under 5 years of age?