QUANTITATIVE DATA INGEST PROCESSING PROCEDURES

UK DATA ARCHIVE
UNIVERSITY OF ESSEX
WIVENHOE PARK
COLCHESTER
ESSEX, CO4 3SQ

Contents

Scope

What is in this guide?

This document describes some of the common data processing checks, techniques and procedures undertaken during the ingest processing of quantitative data collections at the UK Data Archive. Since the bulk of quantitative data acquired by the Archive remains social survey data or similar, the guidance is concentrated on the major statistical software packages and formats encountered, such as SPSS and Stata. Every data collection is unique and this guide is not intended to provide exhaustive information on all elements of ingest processing, which may be found elsewhere:

- Procedures for running the automated quantitative Python processing script within SPSS as reference in this document are contained in the document *Processing Script Procedures. (*This is not currently a controlled document as development of the script is ongoing and the script is not appropriate for certain data collections.)

- Procedures for qualitative data collection ingest processing are contained in the document *Qualitative Data Ingest Processing Procedures.*

- Procedures for the ingest processing of study documentation are contained in the document *Documentation Ingest Processing Procedures.*

- Archive processing standards for data ingest processing are covered in the document *Data Ingest*

Processing Standards.

- For a brief guide to study processing, the document *Ingest Processing Quick Reference* should be consulted.

Please note that some of the documents referenced within this document are not publicly available, but external readers may contact the Archive for advice in case of query.

Secure Access Ingest Processing

Procedures for Secure Access data collection ingest processing are contained in the document *Secure Access Ingest Procedures*. Many of the data checking procedures described below are also applicable to Secure Access quantitative data, but are conducted within an isolated Secure Access processing environment.

ISO 27001 Information Security

Ingest processing at the UK Data Archive, whether Secure Access or not, is undertaken according to the principles of the ISO 27001 Information Security standard. Data security and integrity are of paramount importance.

Roles and Responsibilities

Where applicable, roles and responsibilities are as defined within the document.

1. Quantitative data procedures in SPSS

Introduction

Successful data archiving requires a balance between effective archival storage and the provision of data to users. The former requires a 'pure', cross-platform format in which to store data, preferably ASCII. The latter requires data held in familiar and well-supported software formats to enable easy secondary use.

The 'Statistical Package for Social Sciences' (SPSS) software is well-used within the academic and social science user communities. Thus it is not surprising that the majority of UKDS/Archive depositors deposit quantitative data in SPSS format. Similarly, the majority of UKDS/Archive users choose to receive quantitative data in SPSS format, though Stata is increasing in popularity where sophisticated analysis and weighting is required. The Archive will create SPSS and Stata versions of deposited data files where possible and appropriate. At present, the bulk of Archive ingest processing work is undertaken using SPSS. Note that a dissemination copy of each data file **must** be created and data edits **only** undertaken on the copy, leaving the 'original' deposited file(s) to be archived as received.

Depending on the nature and condition of the individual study, assessment and ingest processing work may include a combination of one or more of any of the checking procedures set out below. These may include the generation of descriptive statistics, a data dictionary and a set of variable frequency distributions. Every study is unique; the content and combination of checking procedures used will depend entirely on the nature of the data in question and the processing standard allocated to the study (see document *Data Ingest Processing Standards*).

Data checking and processing in SPSS is normally conducted using a syntax file in preference to menu-generated commands (which are also available for most common SPSS procedures). Not only is it easier to run multiple procedures using a single syntax file, but the syntax file will provide a record of data edits undertaken that can be archived with the study as appropriate. It can also facilitate easy re-use of complex syntax for future deposits in a data series.

The SPSS routines described below are in alphabetical rather than any preferential order. The routines described here are based on those available in SPSS version 19. If using a different version, please refer to the software help guides accordingly.

1.1. Generating descriptive statistics

In SPSS for Windows, to generate descriptive statistics for all variables, the following command may be used in a syntax window:

    DESC all.

Specimen Descriptives output:

	N	Minimum	Maximum	Mean	Std. Deviation
AGE3 Age	11471	15	65	39.67	12.99
SEX Sex	11471	1	2	1.50	.50
QUOTA Stint number where interview took place	11471	1	221	109.84	65.25
WEEK Week number when interview took place	11471	1	13	7.12	3.69
W1YR Year that address first entered survey	11471	5	5	5.00	.00
QRTR Quarter that address first entered survey	11471	3	3	3.00	.00
ADD Address number on interviewer address list'	11471	1	11	3.56	1.81
Valid N (listwise)	11471				

Greater output can be obtained from the descriptives command using the statistics subcommand:

DESC all
/stats=all.

Selecting the /all subcommand will provide the standard error of the mean, variance, skewness and kurtosis. Note, however, that the default output (minimum, maximum, mean and standard deviation) is normally sufficient for Archive processing purposes.

The descriptives output provides the basis for checking anomalous values, especially for nominal (categorical) variables. For example, consider a nominal (categorical) variable that is (according to the depositor's documentation and/or value labels in the SPSS file) supposed to range between 1 and 8. When one examines the descriptives output, it shows a maximum value of 9. There is, therefore, an additional value that has not been defined by the depositor. Either this represents an undefined code or one or more errors in the data. The frequency distribution of that variable can then be examined to see the extent of the problem.

The descriptives command can also throw up erroneous values for interval variables. In many instances, one will have little knowledge of the possible range of values for an interval variable, but in the case of some variables, such as 'age' (age in years), values below 0 or above, for example, 120 would indicate the presence of possible errors.

1.2. Generating a data dictionary/codebook

The DISPLAY DICTIONARY command

The display dictionary command generates what SPSS calls the 'data dictionary', which is a detailed description of the contents of the data file, giving details of each variable in terms of the variable name, variable label, variable type, format, missing values, and value labels. This command will display the data dictionary for the whole file, and can be modified to display a list of selected variables. see online syntax guide within SPSS for details.

To run the display dictionary command, the following syntax can be used:

DISPLAY DICTIONARY.

The CODEBOOK command

In SPSS versions 18 and over, the syntax command CODEBOOK can also be run. This does not produce a data codebook with frequency distributions, but output that is more of an integrated 'data dictionary', with variable and value labels and other settings listed together in variable order. Subcommands can be set to customize output; see the current version of SPSS in use for details.

Useful elements of the display dictionary/codebook command output

If the descriptives command forms the basis of checking the data, the display dictionary command forms the basis of checking the internal metadata. The most useful elements of data dictionary/codebook output are as follows:

Variable name:

> Lists the name of each variable.

Standard attributes:

> **Variable label**
>
> A description of the variable appears to the right of the variable name; this is the variable label (if one has been defined). Note that SPSS can store variable labels up to 255 characters, but some automated Computer Assisted Personal Interviewing (CAPI) questionnaire software packages   may truncate labels or add random characters, such as @. These labels may need to be edited, especially for A* standard studies that are to be added to Nesstar (see document *Data Processing Standards*).
>
> **Type**
>
> Indicates the variable type. Common types are numeric, string (text) or date format.
>
> **Measurement level**
>
> For numeric variables this can be nominal, ordinal or scale (i.e. interval)[1]. However, this cannot always be taken as meaningful as SPSS defines these using a simple rule based on the number of unique values (calculated on the fly when the file is opened): numeric variables with fewer than 24 unique values and string variables are set to nominal, and numeric variables with 24 or more unique values are set to scale (SPSS settings can be changed if necessary).
>
> **Format (print and write)**
>
> Unless the depositor has specified otherwise, print and write formats will be the same. If the Format is of the form FX (where X is a number, typically 8), the variable is almost certainly nominal (categorical) or ordinal (since no decimal places are defined). If the variable format is of the form FX.Y (where Y is the number of decimal places) then the variable may well be interval (though note that F8.2 is SPSS's default format).

Missing values

> This line only appears if any user-defined missing values are present; system-missing values are not listed.

Valid values

> Describes the value labels that have been defined for a variable. Each value label is listed.

Labelled values

> Where a variable contains a set of numeric values rather than codec categories, such as income, age, etc., some values may be labelled, for example where age is topcoded at 75 and labelled to indicate that: '75' = '75 and over'.

---

[1] Useful definitions of quantitative variable types may be found on the GraphPad.com web site at: http://www.graphpad.com/faq/viewfaq.cfm?faq=1089 (retrieved October 26, 2015).

1.3. Making use of the descriptives and data dictionary/codebook output

In combination with a visual examination of the data, the descriptives and data dictionary/codebook output provides the basis for the following checks:

- unlikely or impossible values for interval variables;
- undefined or incorrect values for nominal (categorical) variables;
- completeness and interpretability of value labels for nominal (categorical) variables;
- missing values appear sensibly and consistently defined (for example, if 'refused' is defined as missing for one variable, is it defined as missing for other variables?).

1.4. Generating frequencies

The SPSS frequencies command provides useful information in addition to that provided by the descriptives command. For example, consider a nominal (categorical) variable that is supposed to range between 1 and 8, but the descriptives output shows a maximum value of 18. From the descriptives output it cannot be seen whether there is a single case with a value of 18 (in which case, it's most probably a data entry error), or whether there are many values between 8 and 18, in which case, a more substantial problem exists (either incorrect mapping of value labels or very 'dirty' data).

To see the number of cases (observations) in every value category the SPSS FREQUENCIES command must be run (the abbreviation 'FRE' may be used):

FRE all

- generates frequencies for all variables. (Note that SPSS may have an internal limit on the amount of variables for which frequencies may be generated in one command; this is typically 1,000. in these cases, separate frequency statements that respect these limits will need to be written, using the 'to' command, as given in the example below.)

FRE var1 var2 var3 var4

- generates frequencies for the variables specified (substitute actual variable names for var1, var2, etc.).

FRE var1 to var321

- generates frequencies all variables that lie between var1 and var321 in SPSS file order (substitute actual variable names for var1 and var321).

Useful frequencies subcommands

To generate frequencies for nominal (categorical) variables only, the formats limit sub-command can be used to suppress output for variables that have more than a specified number of values. In the example below, only frequencies for variables with up to 30 unique values will be generated - which will suppress output for the interval variable age (which has more than 30 unique values), and generates output only for the variable sex, which has two unique values.

FRE age sex
/format limit=30.

Specimen output

SEX					
	sex	Frequency	Percent	Valid Percent	Cumulative Percent
Valid	1 Male	5750	50.1	50.1	50.1
	2 Female	5721	49.9	49.9	100.0
	Total	11471	100.0	100.0	

1.5. Checking label metadata

The DISPLAY LABELS command generates variable labels. These are also generated by the DISPLAY

DICTIONARY command (along with value labels and variable format information), but to simply generate variable names with their SPSS data file position and variable labels, use the DISPLAY LABELS command.

DISPLAY LABELS.

Specimen output:

Variable Labels		
Variable	Position	Label
AGE3	1	Age
SEX	2	Sex
QUOTA	3	Stint number where interview took place
WEEK	4	Week number when interview took place
W1YR	5	Year that address first entered survey
QRTR	6	Quarter that address first entered survey
ADD	7	Address number on interviewer address list

Display labels can be used as a quick guide to highlight which variables have no label. This may be useful for studies to be processed to A/A* standard, where missing variable labels are generally added.

1.6. Weighted studies

Many studies contain variables that are used to perform weighted statistical analysis (a technique typically used to make a sample representative of some important criteria, such as population figures). However, it is desirable to take off the weighting for ingest processing and archiving, unless the depositor has specifically requested that the weight remain on (this should be recorded in the Note file for information).

If an SPSS system data file (.sav format) has the weights set on, 'Weight on' will be displayed in the bottom right hand corner of the SPSS data window. However, this does not always display when the SPSS file is in 'portable' format (.por), so the syntax command to show whether the study is weighted is:

SHOW WEIGHT.

For unweighted data, the output should display 'File is not weighted'

If the output reads 'weighting variable = <variable name>', the data are currently weighted by the variable shown. Weighting data will not alter the core data matrix (i.e. the actual data values SPSS displays in its data viewer) but will alter any statistical output, such as descriptive statistics. The weight should therefore be removed, using the command:

WEIGHT OFF.

Documentation should be checked to ensure that comprehensive information on weighting variables and their construction and use is given. The depositor should be contacted and extra information requested if that provided is not adequate.

For a useful guide to the principles of data weighting, see the UK Data Service guide, *What is weighting?* at http://ukdataservice.ac.uk/media/285227/weighting.pdf (retrieved November 17, 2014).

1.7. Identifying cases with anomalous values

The commands described above, particularly descriptives and frequencies, will show any anomalous values, particularly for nominal (categorical) variables. An anomalous value of a nominal variable is one that is unexpected (according to the documentation and/or internal metadata). For example, the documentation and value labels in SPSS for the variable mstatus (marital status) may be described both in the documentation and the SPSS value labels as 1=married, 2 = divorced, 3 = single; yet the descriptives command reveals a maximum value of 6, and the frequencies command reveals a substantial number of cases of values 4 and 5 as well. As a result, the meaning of codes 4-6 is unknown. They may either be invalid (i.e. data errors) or they may be incorrectly labelled (i.e. the data are correct but the codes 4 to 6 have not been defined by value labels or in the documentation).

Assessment of an anomalous value for an interval variable may rest on logic and knowledge alone, rather than whether or not a value label or code is defined. For variables that are inherently comprehensible – like age or income – checks can be made for values that lie outside the realm of probability or possibility (such as less than 0 or greater than 120 for an age variable).

Whenever an anomalous value is found, the offending case(s) may be examined to see if the other variables for that case provide any clue as to the source of errors. To do this, the Temporary command followed by the Select If command should be used. A logical statement must be provided that selects out cases that meet a given condition, such as a certain variable having a certain value. The Select If command should **always** be preceded by the Temporary command, otherwise a permanent selection of cases will be made, and if the file is saved all non-selected cases will be lost. (For safety reasons, processing work must always be undertaken on a processing copy of the file rather than the original.)

The following example temporarily selects all cases for which the variable Mstatus is greater than 3 (for example, values may be 1=married, 2 = divorced, 3 = single). The LIST command will then provide the number/name of any variable(s) specified. In the example below, the Hhno (household number variable) of all cases that record an Mstatus value in excess of 3 will also be given alongside the value of Mstatus.

```
TEMP.
SELECT IF mstatus>3.
LIST mstatus hhno.
```

Using the value of the Hhno variable (or whatever the data file's unique identifier variable may be), the cases in the data file with an anomalous Mstatus value can be examined to see if there is any discernible reason for the anomaly. There may be some distinct pattern, for example, all cases in a certain geographical area contain an undefined code for a location variable, or all people aged under 16 display an undefined marital status variable. Even if such detective work does not provide a definitive answer by itself, the more information that can be discerned during processing, the more likely a speedy solution can be sought from the depositor, or information provided for the secondary user.

1.8. Print and write formats, variable width and decimal places

SPSS print formats control how variables are displayed on the screen in the SPSS data viewer (the 'variable width' column in variable view). The default format for a numeric variable is defined as **F8.2**, which allows up to 8 digits to the left of the decimal point and two digits to the right (i.e. two decimal places). An **F8.2** format will display numbers up to 99999999.99 and will use scientific notation (E+X) for larger numbers.

Print and write formats can be problematic when importing data into SPSS. In some instances, variables with many decimal places may be displayed as an integer (e.g. 1.348726 being displayed as 1), or with only one or two decimal places. This may confuse naïve end users of the data, though the data are correct. More importantly however, print and write formats should also be considered as data export formats, as they can affect certain conversions, most importantly from SPSS portable (.por) to Stata, whereby the resulting data file only holds the data to the number of decimal places given in the SPSS print format, rather than the number contained in the SPSS data file itself. For safety, SPSS to Stata conversions should always be performed using an SPSS system format file (.sav) rather than portable format. The Python processing script currently in use at the Archive creates Stata from .sav format within SPSS.

The variable width in SPSS can present similar problems. A value of 10 or -1 can exist in a variable defined as having a width of 1 (e.g. defined as F1.0), but this can also affect export formats, depending on how conversion is performed. For example, two-character missing values (e.g. -1) will display as * if the format is set to F1.0. This can create problems for both Nesstar and Stata conversion, so all quantitative data files should have the formats for variables set to the appropriate display format length before the processing script is run.

If such formats inevitably lead to rounding, truncation or loss of data upon conversion to other preservation or dissemination formats, they must be altered prior to conversion. Therefore, all file transfers must be checked thoroughly to ensure data integrity is preserved.

The SPSS command to change or set formats is as follows:

FORMATS age sex marstat (F2.0).

This command will set both print and write formats for the numeric variables 'age', 'sex' and 'marstat' to two characters, with no decimal places.

FORMATS income (F10.6).

… will set both print and write formats for the numeric variable 'income' to 10 characters and 6 decimal places.

FORMATS council (A20).

… will set both print and write formats for the string (text) variable 'council' to 20 characters.

For further information on formats and similar commands, see SPSS online help or manuals.

1.9. Adding variable and value labels to data in SPSS

One of the most common processing tasks undertaken at the Archive is the addition/editing of metadata, such as variable and value labels. The extent to which this is undertaken depends on the condition of the deposited data file (i.e. how many labels are missing) and the Archive processing standard allocated to the study (see document *Data Ingest Processing Standards*). Labelling is always carried out on the dissemination copy of the data file, not the deposited 'original' (see section 1 above), and is usually carried out prior to data format conversion (see section 4 below). In order to enable 'tracking' of the edits undertaken, the labels are added using a syntax file rather than being added directly to the file via the SPSS graphical interface. The syntax file is then archived with the study. The archiving of syntax files also has another advantage; in the case of series studies, the same labels are often missing or need editing at each wave. Therefore, archived syntax from the previous wave may often need only minor editing to enable it to run on the latest wave, reducing subsequent work. Of course, thorough checks should be made against the latest wave's data and documentation to ensure that the labels are still valid.

When variable and value labels have been added/edited as necessary, the results should be checked by running frequencies to check the amended variables. If the additions/edits have been successful and all other errors in the data fixed or noted, conversion to alternative dissemination, and archival formats – such as Stata - may be undertaken (see below).

Adding variable labels

To add/edit variable labels in SPSS, open the data file required, and a syntax file. Use the 'variable labels' command, and list underneath the variables that require labelling alongside the label required (see example below). At the end of the list, use a full stop as the command terminator.

Adding/editing variable labels

```
VARIABLE LABELS

nb7pint      Pint equivalent
fuh          Family unit head
fuhage       Family unit head age
fuhsex       Family unit head sex
husbage      Age in years of male partner
husband      Person number of male partner
husbmar      Marital status of male partner
mothage      Age in years of mother
mother       Person number of mother
partage      Age in years of partner
```

.

Note: where the label contains a character that SPSS usually interprets as a command, e.g. / or parentheses , the label should be enclosed in single quotation marks. For example,

hiqul11d   'Highest qualification (detailed grouping)'

Where the required label contains a single quotation mark, the label will need to be enclosed in double quotation marks:

Lastpaydk   "Value of last payment: don't know"

Once labelling is complete, the syntax file should be saved and archived with the study in the dp/code/spss (or appropriate software format) directory. For practical purposes, variable/value label and other edits may be combined into one syntax file.

Adding value labels

To add/edit value labels in SPSS, open the data file required, and then open a syntax file. The procedure is similar to that used for variable labelling, with the variable names and required values specified (see example below). Two SPSS commands may be used: 'add value labels' or 'value labels':

- 'add value labels' will add labels or change only those specified in the syntax. This option is often more appropriate for Archive processing, as in many cases, only one or two value labels from an individual variable may be truncated or missing, while others are correct.

- 'value labels' will erase all existing value labels within the variable and add/edit only those specified in the syntax. Therefore, to provide adequate value labelling, the whole set of value labels for the variable must be specified.

Again, where the label contains a character that SPSS usually interprets as a command, e.g. / or \ , the label should be enclosed in single quotation marks. Where the required label contains a single quotation mark, the label will need to be enclosed in double quotation marks

Similarly, to label 'string' or textual variables, single quotation marks may need to be used. If unsure, refer to the help guide for the version of SPSS in use.

```
ADD VALUE LABELS
grifuh grifp grhheq ntquint pfempgr1 traingr1
1       Yes
2       No
-9      Does not apply
-8      "Don't know"
-7      Refused section
-6      'CHILD/MS/PROXY'
.
```

Once labelling is complete, the syntax file should be saved and archived with the study in the dp/code/spss (or appropriate software format) directory. As described above, the variable/value label and other edits may be combined into one syntax file.

1.10. Converting from SPSS to Stata

The conversion process from SPSS to the other current Archive standard dissemination formats (Stata and tab-delimited text) is currently undertaken on SPSS system files (.sav), using an automated Python script that runs in SPSS. The procedure for installing and using the script is contained within a separate document. Once the dissemination formats have been created, they should be checked using analogous procedures to the SPSS checks already undertaken, to ensure data integrity during transfer. Obviously, the checks undertaken on tab-delimited files are more limited than those that can be undertaken in Stata.

Checking SPSS to Stata transfers

Transfer between SPSS and Stata can sometimes be problematic due to the differing nature of the two software packages. Thorough checks must be carried out on all Stata files generated from SPSS, either via the Archive Python processing script, via proprietary software such as StatTransfer, or exported direct from

SPSS (versions 14 and above). For example, string variables are treated differently in Stata (though less so in the most recent versions of Stata) and allowed variable and value labels lengths may differ. Most problems are evident from very basic post-transfer checks, but not all; information on more complex problems that may result during transfer (and their solutions) are available internally for Archive staff, covering incorrect value transposition, incorrect missing value transfer and problems with variable name transfer. The data checking procedures that are most closely analogous to those usually carried out in SPSS are listed below.

2.   Data checking procedures in Stata

The Stata statistical software package has a command language similar to that of SPSS. Stata is currently the second most popular statistical software package among the Archive's user community. It is growing in popularity due to its ability to perform sophisticated statistical tests and is preferred by many economists. Some of the key Stata commands useful for Archive data processing are detailed below.

2.1.   Analogous Stata command to 'Descriptives'

The closest Stata command to the SPSS Descriptives command is the Stata **summarize** command. This will produce descriptive statistics for all numeric variables, displaying: number of (valid, i.e. non-missing) observations; mean; standard deviation; and minimum and maximum values.

Specimen output from the Stata **summarize** command:

Variable	Obs	Mean	Std. Dev.	Min	Max
state	0				
region	50	2.66	1.061574	1	4
pop	50	4518149	4715038	401851	2.37e+07
poplt5	50	326277.8	331585	35998	1708400
pop5_17	50	945951.6	959373	91796	4680558
pop18p	50	3245920	3430531	271106	1.73e+07
pop65p	50	509502.8	538932	11547	2414250
popurban	50	3328253	4090178	172735	2.16e+07
medage	50	29.54	1.693445	24.2	34.7
death	50	39474.26	41742.35	1604	186428
marriage	50	47701.4	45130.42	4437	210864
divorce	45	23679.44	25094.01	2142	133541

Note: 'State' is a string variable in the example above, so Stata cannot give any output. Also, the 'divorce' variable must have five missing cases, as valid observations are given as 45 rather than 50.

Like SPSS, individual variables can be specified. For example, the command:

summarize death marriage

… will only generate descriptive statistics for these two variables.

Unlike SPSS and SAS, Stata commands are case sensitive (they are always lower case). The wildcard (*) can also be used for most Stata commands, for example:

summarize a* m*

will summarise all variables beginning with the letters a or m.

2.2.   Analogous Stata command to 'Display Dictionary/Codebook'

There are two choices in Stata. As detailed below, the **describe** command produces output that is slightly less detailed than the SPSS data dictionary, while the **codebook** command produces output that is more detailed. These two options are discussed in turn:

The **describe** command generates variable names, storage formats (using the Stata definition, not SPSS formats); display formats (Stata rather than SPSS); value labels and variable labels.

Specimen output from the describe **command:**

```
obs:            50              1980 Census data by state
vars:           12              6 Jul 2000 17:06
size:           3,000 (99.5% of memory free)


variable name   storage type    display format    value label    variable label

state           str14           %-14s                            State
region          int             %63.0g            test           Census region
pop             long            %12.0gc                          Population
poplt5          long            %12.0gc                          Pop, < 5 year
pop5_17         long            %12.0gc                          Pop, 5 to 17 years
pop18p          long            %12.0gc                          Pop, 18 and older
pop65p          long            %12.0gc                          Pop, 65 and older
popurban        long            %12.0gc                          Urban population
medage          float           %9.2f                            Median age
death           long            %12.0gc                          Number of deaths
marriage        long            %12.0gc                          Number of marriages
divorce         long            %12.0gc                          Number of divorces
highdiv         int             %63.0g            yesno          High divorce state?
```

2.3. Guide to understanding Stata storage types:

str

string (text variable, of specified length); e.g. in the example above, 'state' is a string variable with 14 characters (str14).

int and **long**

integer variables, i.e. a categorical or ordinal variable; see 'region' above.

float and **double**

an interval (continuous) variable of up to 8 digits in accuracy, see 'medage' above.

Note also that value labels have only been defined for the highdiv variable. Yesno is the name of the value label (this is not the same as a 'value label' in SPSS; this feature does not exist there). The value labels are mapped to one or more variables sharing the same coding (e.g. several variables might have the 'yesno' value label.

To look at value labels (as defined in SPSS) themselves, use the **label list** command, e.g.:

Label list highdiv

To generate output that is more detailed than the SPSS data dictionary, use the **codebook** command:

Specimen output from the codebook **command:**

```
state ----------------------------------------------------------------- State
type: string (str14), but longest is str13
unique values:  50            coded missing:  0 / 50
examples:  "Georgia"
```

```
"Maryland"
"Nevada"
"S. Carolina"
warning:  variable has embedded blanks
region ---------------------------------------------------- Census region
type:  numeric (int)
label:  regcode, but 3 values are not labelled
range:  [1,4]                 units:  1
unique values:  4            coded missing:  0 / 50
tabulation:     Freq.    Numeric  Label
9               1
12              2          Tendring
16              3
13              4
pop      ------------------------------------------------------------- Population
type:  numeric (long)
range:  [401851,23667902]        units:  1
unique values:  50           coded missing:  0 / 50
mean:  4.5e+06
std. dev:  4.7e+06
percentiles:     10%          25%          50%          75%          90%
                 67174        1.1e+06      3.1e+06      5.5e+06      1.1e+07
```

As can be seen, this gives extremely detailed information for each variable, almost as much as descriptives and the data dictionary combined in SPSS terms.

3.   Checking tab-delimited text files

The Archive's current non-proprietary dissemination format is tab-delimited text (either of the ASCII or UNICODE character set), though it is envisaged that this may move to XML (extensible markup language) in time.

The limitation of tab-delimited format is the fact that it can only store the rectangular matrix of data points and variable names. What can be termed internal metadata - variable descriptions (labels), code descriptions (value labels), and whether certain codes are defined as 'missing', cannot be stored in the same file as the data. Such information needs to be stored in an additional series of tab-delimited files or in the documentation.

Tab-delimited text can be read into almost any statistical package, database, spreadsheet or word-processor, and so can also be considered a dissemination format, though most users request data in a software dependent format. Similarly, it is rare nowadays for data to be deposited in tab-delimited or other text format, unless in the form of database output files when no other alternative is possible.

Therefore, tab-delimited text will be encountered most frequently during checks on the outputs created by the Python processing script currently in use at the Archive. The following checks should be made, using the original SPSS .sav file and RTF data dictionary file (created by the script and archived under mrdoc/allissue) as a guide:

- all cases have transferred successfully
- the variable names are included in the top line of the tab-delimited file
- all variables have transferred successfully.

Note that the visual inspection of tab-delimited files during output checking may be more successful with a simple text-editing software package such as Pfe32 (or even NotePad for small files). More sophisticated text-editing software packages such as UltraEdit may wrap long lines and make it difficult to assess whether the correct number of cases has been transferred by SPSS.

4.   Rounding

All file transfers, whatever the source or destination format, should be checked thoroughly for rounding of decimal places. This can affect Microsoft Excel display formats, when files are exported for use in other packages. Depositor-specified limitation of decimal places (to make the data or output more easily interpretable) should be treated as a data export format. Export to, for example, tab-delimited text, will result in data rounded to the displayed number of decimal places rather than the number in the underlying data. Similarly, export of tables from Microsoft Access can result in rounding of decimal places to two digits, due to the internal software settings. See below for information on the processing of Access databases.

5. Data checking procedures in SAS

Occasionally, studies may be deposited in SAS format, although this is rare nowadays. SAS also has a full command language that allows similar (and in some areas greater) functionality to SPSS and Stata. Data deposited as .sd7 or .sasb7dat files can be read directly into recent versions of SPSS. SAS transport files (.ctp, .xpt) need to be converted using SAS 8.0 for Windows or above.

Note that the transfer of some data files from SPSS to SAS may also be problematic. Even filenames may need to be changed due to SAS's more specific requirements. Checks should be carried out accordingly.

SAS version 9.0 onwards will import SPSS .sav files directly into SAS, using the SAS 'Import Data' facility. Users who request SAS format should be advised to obtain the SPSS version of the study required and try this before requesting SAS files to be generated by the Ingest team.

6.   Access databases

Microsoft Access is the most common ingest format for databases. It should be noted that each version of Access has some incompatibilities with former versions. Access databases may comprise tables only, or those with some functionality, such as formatting, queries, forms, reports and/or macros. For table-only Access databases, an Excel version may be created alongside the Access database, as an alternative dissemination format.

Where functionality exists, the tools (macros, reports etc.) should be examined to ascertain whether they are an aid to secondary analysis. If so, the usual dissemination format must remain an Access database.

The Archive preservation format for Access is almost always a tab-delimited version of each table, since Access databases may contain long (>255 character) textual strings with commas included (hence CSV output may not be appropriate.

Deposit in Access format is currently more common for historical databases and some qualitative and Rural Economy and Land Use Programme (RELU) data.

The following procedures should be read carefully before any conversion is attempted on an Access database.

6.1. Memo fields

Memo fields in Access 2010 allow '1 gigabyte of characters', of which 65,535 characters can be displayed in a text box. However, versions of access prior to Access 2010 should be treated with care, as they may only allow textual strings of >255 characters in 'Memo' fields. They may also contain embedded special characters such as tabs and carriage returns. Memo fields must be carefully checked for two reasons:

- When exported to tab-delimited text format, memo fields may be truncated when subsequently imported into other packages (including import back into Access).
- Where embedded tabs or carriage returns exist, data may not import as a rectangular matrix.

To ascertain whether a table within Access contains memo fields, right-click on the table name, and choose 'Design view' from the list. This will show how each field is defined in Access. If there are no memo fields, conversion to tab-delimited format may proceed. To establish whether embedded carriage returns or tabs exist in the memo fields the table concerned can be test-converted to tab-delimited format and reopened in

Access (see below). If the data are no longer in the same rectangular matrix (i.e. the rows and columns no longer tally), they probably contain embedded tabs or carriage returns. To confirm this, the contents of cells can be cut and pasted into a text editor, or Word, to examine for embedded characters. If embedded carriage returns and/or tabs are found to exist, they should be removed. Before any editing work is undertaken, a copy should be made of the Access database, and the editing work undertaken on the copy, to avoid any inadvertent damage.

Once the embedded carriage returns and tabs have been removed, the Access table should convert successfully to tab-delimited format. The long strings may need further checks when the tab-delimited version is converted to another format.

6.2. Converting Access tables to tab-delimited format

Following depositor and user feedback, the Access database should remain the primary dissemination format, but the tab-delimited text files generated via the procedures described below make a useful archival format, and should be archived under noissue. The tab files may of course be supplied to users on request.

6.3. Decimal place rounding during table export from Access

Exporting Access tables directly into tab-delimited text format may automatically round/truncate all values to two decimal places. This can still be a problem with Access 2010.Therefore, if a table within the database contains values with more than two decimal places, a few extra steps should be undertaken to ensure this truncation does not occur.

Firstly, potential problems may be averted by the export of individual tables to MS Excel (first check the decimal place setting in MS Excel is not set to round to two decimal places). This works for some Access databases, but not for others. If export to Excel works well, the resulting files may then be exported from Excel to tab-delimited text, taking care to ensure no further rounding/truncation occurs in that transfer.

If export to Excel is not successful in avoiding rounding, a copy should be made of the Access database, as for the editing of memo fields described above. As changes are to be made to table construction, performing the work on a copy of the database is essential, to avoid passing changes on to secondary users of the Access database.

1. In Access, select the table for export, then click on the 'Design' button to open the Design view of the table.
2. Each Field Name should have a Data Type specified. To prevent decimal place truncation, the Data Type of the affected variable(s) should be changed to Text.
3. Click on the current Data Type and a drop-down menu should appear. Text should be selected.
4. Save the database.
5. The table can then be closed which will return to the Main view where all tables within the database may be seen.

6.4. Converting multiple Access tables to CSV or comma-delimited text format

This can be done via StatTransfer's command processor – all tables can be exported at once to separate files at once using the '-t*' command. However, the resulting files should be examined carefully in case of problems such as rounding, in case settings need to be changed. Also, if some tables in the database are set as 'read only', this method will not work.

6.5. Converting Access tables from within the Access database

Access 2010

To convert to tab-delimited format, open the database. The tables should be listed on the left-hand side of the screen, under 'All Access Objects'. Right-click on the table name required and select **Export > Text File**. Don't select an option from the list, just click OK. In the next screen, leave the 'Windows (default)' option

selected and click OK. From the next screen, select the **Delimited** option and click **Next**. Under 'Choose the delimiter that separates your fields', select **Tab** (if you want to create a CSV file, you can select 'Comma' here instead), tick **Include Field Names on First Row**, and select **{None}** from the 'Text Qualifier' drop-down menu, then click **Next.** Finally, choose the appropriate file destination and change the file extension from .txt to .tab (or .csv if appropriate) and click **Finish**.

Tables can be exported directly into Excel (see also below), but there are some limitations; for example date values earlier than Jan 1, 1900 are not exported The corresponding cells in the worksheet will contain a null value. This may be problematic for historical databases. Also, the number of cases may be affected by Excel's limitations, meaning incomplete export.

Older versions of Access

Access 2007

To convert to tab-delimited format, open the database and select the table to be exported. Right-click on the table and select **Export > Text File**. Do not tick an option from the list, just click on OK. In the next screen, select the **Delimited** option and click **Next**. Under 'Choose the delimiter that separates your fields', select **Tab** and tick **Include Field Names on First Row**, and select **{None}** from the 'Text Qualifier' drop-down menu, then click **Next.** Finally, choose the appropriate file destination and change the file extension from .txt to .tab, and click **Finish**.

Tables can be exported directly into Excel (see also below), but there are some limitations; for example date values earlier than Jan 1, 1900 are not exported The corresponding cells in the worksheet will contain a null value. This may be problematic for historical databases. Also, the number of cases may be affected by Excel's limitations, meaning incomplete export.

Access 2003 and 2002

Each table must be saved separately. To convert a table to tab-delimited text format, select it and then go to:

File Menu> **Export**

Select the appropriate location, then choose **Text Files (*.txt; *.csv; *tab;*asc)** from the drop-down menu. Click Export, then in the **Export Text Wizard** interface, make sure **Delimited** is selected, then click **Next**, then do the following:

Choose the delimiter that separates the fields: **tab; then** tick **Include Field Names on First Row; s**et **Text Qualifier** to **{none}, t**hen click on **Next**, then **Finish**.

You may need to change the resulting file extension from .txt to .tab.

6.6. Converting Access tables to MS Excel format

Access tables may also be converted into Excel format for dissemination, but there are some limitations; for example date values earlier than Jan 1, 1900 are not exported The corresponding cells in the worksheet will contain a null value. This may be problematic for historical databases. Also, the number of cases may be affected by Excel's limitations, meaning incomplete export.

As noted above, export can be done interactively between MS Access 2010 and Excel 2010. Care should be taken to avoid similar rounding and truncation of values. Export of all tables to individual Excel files can be done as a batch process using Stat Transfer 11; refer to the help guide for Stat Transfer for details.

For earlier versions of Access and Excel, or where transfer is to be done from within the Access database without using Stat Transfer, the following issues must be borne in mind. When long text strings contain a certain combination of numeric characters, for versions prior to 2007 and 2010, Excel cells may be truncated at 255 characters without warning. Some cells in a given field may be truncated, but not others. Therefore, extra checks should be undertaken on all fields, and to avoid further truncation problems with long text strings, all fields that were originally memo fields in Access must be assigned as text fields in Design View before export. Excel text fields, unlike Access, may be over 255 characters in length. If this is the case, a copy of the Access database must be made before any changes are saved, as for the avoidance of decimal place rounding, described in section 6.3 above.

6.7. 'Documenting' the Access database and archiving the data structure

Documenting variables

The Access 'documenter' command should be run for every table. This may be done as follows:

In Access 2010

- In the Database Tools tab, select **Database Documenter**

- In the pop-up **Documenter** window, click **Select All**

- Under **Options**, in the **Print Table Definition** window, make sure:

    o **Properties** and **Relationships** boxes are ticked under **Include for Table**

    o **Names, Data Types and Sizes** is selected under **Include for Fields**

    o **Names and Fields** is selected under **Include for Indexes**

- Click on **OK**, and then **OK** again in the **Database Documenter** window

- The **Print Preview** window will open; using the buttons at the bottom of the window, you can scroll through the table documents. Once the pages have been checked, using the **Print** button, an Adobe PDF file (remember to make sure all pages are printed) can be created and named appropriately, e.g. NNNN_accessdatabase_document.pdf (where NNNN is the study number).

In Access 2003 and earlier versions:

- Under the **Tools** menu select **Analyze** and **Documenter.**

- Select one table at a time (unless there are many tens of tables, in which case it is acceptable to select all and thereby generate and archive one block of output that covers all the tables).

- To control the level of output select **Options.** The following elements should be chosen:

    o **Properties**

    o **Relationships**

    o **Names data types, size and types**

    o **Names and fields**

- The files (one for each data table converted to tab-delimited text format) should be exported in RTF format and be named 'NNNN_<table name>_variableinformation.rtf' (e.g. 5662_<table name>_variableinformation.rtf). They should be archived under mrdoc/allissue

6.8. Documenting table relationships

Larger Access databases may contain a complex network of relationships between tables. A diagram of the table relationships may be created, both to aid users (especially those who request the tab-delimited files) and as a preservation tool. Therefore, for those Access databases that need it, the table relationships may be documented as follows:

In Access 2010:

- In the **Database Tools** tab, select **Relationships**

- In the **Show Table** box, highlight all tables in the list to select them and click **Add**

- Once the tables show in the **Relationships** window, click on **Close** in the **Show Table** box.

- In the **Relationship Tools** top menu, click on **All Relationships**. If table relationships exist (they don't in all databases) they will be shown.

- Click on **Relationship Report** to display a print preview screen which will display the table relationships. The **Print** facility can then be used to save the file in Adobe PDF file to be saved with the documentation. The file should be named as NNNN_access_table_relationships.pdf .

In Access 2003 and earlier versions:

- From the Access database top menu, select **Tools** > **Relationships**: a diagram of the table relationships will be displayed onscreen.

- From the top menu, select **File** > **Print Relationships**

- Choose the Adobe PDF printer from your list of printer options, to generate an Adobe PDF file. The file may be named as follows: NNNN_access_table_relationships.pdf , and saved with the documentation. (Note that some image editing may need to be done if the table relationships are extensive, as they may print across more than one page within the file).

6.9. Conversion of tab-delimited files into Access

Although rarely done during processing, conversion from tab-delimited format into Access is also possible. However, it should be noted that Access examines the first few cases in any given field and uses these to define the field on importation. For example, the first few cases may all contain entries of less than 256 characters, even if some of the remaining cases contain longer entries. Because it does not assess beyond these cases, Access will import the entire field as a 'text' field instead of a memo field. This may lead to truncation of the longer records at 255 characters. However, where Access encounters any errors (including truncation), the import process will be aborted and import errors written to a table in the current database. This can then be opened to discover the problem. Import is easier from Excel 2010 into Access 2010, but care should still be taken to ensure that data are not lost.