SAS 5 – Getting Data into SAS

SAS Data Files

- SAS Binary data files
  - .sas7bdat is the current format. These files work cross-platform in both Windows and Unix environments
  - .sd2 and .sd7 are older, Windows-only data files
  - PROC CPORT produces special binary SAS files that can be CIMPORT-ed on any OS.
- SAS ASCII data files
  - XPORT engine files produce ASCII SAS files that can be imported into SAS on any OS.

Data Conversion Software

- If you are on an SSCC computer and want to convert data from some binary file (Stata, SPSS, Excel) to SAS
  - Stat/Transfer
  - DBMS/Copy

SAS Built-in Conversion

- SAS itself has the capacity to convert some foreign data files
  - Library engines (SPSS)
  - `PROC IMPORT` (Excel)
  - GUI Data Import (Excel, comma-separated values)

Text Data Documentation

- Base SAS
  - SAS Language Reference: Concepts
  - DATA Step Concepts
  - Reading Raw Data
- Base SAS
  - SAS 9.2 Language Reference: Dictionary
  - Dictionary of Language Elements
  - Statements
    - `INFILE`
    - `INPUT`

Text Data Overview

- Sources
  - In-line, from a file, or from a URL
- Lines to read/skip
- Observation delimiter(s)
  - End of line, m lines, n data values, or some combination
- Value delimiter(s)
  - Spaces, commas, columns, relative position

Text Data Overview (cont.)

- Standard data values or formatted data
  - Numbers with commas, dollar signs, dates, and times all require informats
- Text qualifiers
  - Can a data value delimiter sometimes be valid as character data? (e.g. spaces or commas)
- Missing values
  - Multiple delimiters, blank columns, special characters, special numerical values

Text Data Sources: in-line

- In-line data

```
data club1;
   input idno name $ team $ strtwght endwght;
datalines;
1023 David red 189 165
1049 Amelia yellow 145 124
1219 Alan red 210 192
1246 Ravi yellow 194 177
1078 Ashley red 127 118
1221 Jim yellow 220 .
;
```

- Nice for examples, you see this a lot in documentation

Text Data Sources: files

- Input text from a file

```
data club2;
    infile "y:\sas\text data\weight loss club.txt";
    input idno name $ team $ strtwght endwght;
run;
```

- This is what you see the most of in practice. It is always a good idea to look at your text file with a text editor/word processor before you write your DATA step.

Text Data Sources: URLs

- Input text from a file on the Internet

```
filename wlclub url
   "http://www.ssc.wisc.edu/~hemken/SASworkshops/data/weightlossclub.txt";
data club3;
   infile wlclub;
   input idno name $ team $ strtwght endwght;
   weightchange = endwght - strtwght;
   run;
```

- Nice for examples/teaching

One observation per line

```
data weight;
   input PatientID $ Week1 Week8 Week16;
   loss=Week1-Week16;
datalines;
2477 195 177 163
2431 220 213 198
2456 173 166 155
2412 135 125 116
;
```

- This is a very typical layout for text files.

Several observation per line

```
data weight;
   input PatientID $ Week1 Week8 Week16 @@;
   loss=Week1-Week16;
datalines;
2477 195 177 163 2431 220 213 198
2456 173 166 155 2412 135 125 116
;
```

- This is common in examples (in the documentation).

Several lines per observation

```
data weight;
   input PatientID $ Week1 Week8 Week16;
   loss=Week1-Week16;
datalines;
2477
195 177 163
2431
220 213 198
2456
173 166 155
2412
135 125 116
;
```

- “flowover”

Several lines per observation (cont)

```
data weight;
    input PatientID $;
    input Week1 Week8 Week16;
    loss=Week1-Week16;
datalines;
2477
195 177 163
2431
220 213 198
2456
173 166 155
2412
135 125 116
;
```

- Multiple input statements

Value delimiters

- List data (space delimited)

```
Lucky 2.3 1.9 . 3.0
Spot 4.6 2.5 3.1 .5
Tubs 7.1 . . 3.8
Hop 4.5 3.2 1.9 2.6
Noisy 3.8 1.3 1.8 1.5
Winner 5.7 . . .
```

- Fixed data (column delimited)

```
Lucky 2.31.9   3.0
Spot  4.62.53.1 .5
Tubs  7.1      3.8
Hop   4.53.21.92.6
Noisy 3.81.31.81.5
Winner5.7
```

List data

```
data toads;
    input name $ hop1 hop2 hop3 hop4;
datalines;
Lucky 2.3 1.9 . 3.0
Spot 4.6 2.5 3.1 .5
Tubs 7.1 . . 3.8
Hop 4.5 3.2 1.9 2.6
Noisy 3.8 1.3 1.8 1.5
Winner 5.7 . . .
;
```

- Character variable names are followed with “$”, otherwise numeric is the default type
- Missing values usually require some sort of placeholder

Fixed data

```
data toads2;
   input name $ 1-6 hop1 7-9 hop2 10-12 hop3 13-15 hop4 16-18;
datalines;
Lucky 2.31.9   3.0
Spot  4.62.53.1 .5
Tubs  7.1      3.8
Hop   4.53.21.92.6
Noisy 3.81.31.81.5
Winner5.7
;
```

- Names of character variables are followed with “$”
- Missing data values can just be blank (spaces)

Qualifiers and delimiters

- Sometimes a delimiter is also valid as part of a data value

```
John Garcia     114  Maple Ave.
Sylvia Chung   1302  Washington Drive
Martha Newton    45  S.E. 14th St.
```

- With character value qualifiers

```
“John Garcia”     114  “Maple Ave.”
“Sylvia Chung”   1302  “Washington Drive”
“Martha Newton”    45  “S.E. 14th St.”
```