
Using SAS[®] to Parse External Data

Andrew T. Kuligowski
FCGI Insurance Group

Using SAS® to Parse External Data

Introduction

Introduction

Parsers – A Primer (*a quick primer*)

Identifying Unique Character Strings

Input Statement – Named Input

Input Statement – Line & Column pointers

Useful Character Functions

Looping & Conditional Execution

Putting it all together

Warning !

The Future (?)

Conclusion

Using SAS® to Parse External Data

Parsers – a primer

For our purposes:

parse

The analysis of a string of characters and subsequent breakdown into a group of components.

Using SAS® to Parse External Data Parsers – a primer

EXAMPLE 1 : *Using a known character string as a guidepost through a document.*

```
BROWSE - USERID.SASRUN.LISTING ----- CHARS ' Cary' FOUND
COMMAND ===>                                SCROLL ===> CSR
NOTE: Copyright (c) 2002-2003 by SAS Institute Inc.,
      Cary, NC, USA.
NOTE: SAS (r) 9.1 (TS1M3)
      Licensed to COMPANY NAME, Site SITENUM.
NOTE: This session is executing on the z/OS V01R07M00
      platform.
```

... ..

Using SAS® to Parse External Data Parsers – a primer

EXAMPLE 1 : *Using a known character string as a guidepost through a document.*

```
BROWSE - USERID.SASRUN.LISTING ----- CHARS ' Cary' FOUND
COMMAND ==>>>                                SCROLL ==>>> CSR
NOTE: The address space has used a maximum of 620K below
NOTE: SAS Institute Inc., SAS Campus Drive, Cary, NC USA
1                                             The SAS System
          Obs      Index      Random1      Random2
          1         1         0.79940         90
          2         2         0.58974         53
          ...         ...         ...
```

Using SAS® to Parse External Data Identifying Unique Character Strings

Now, let's deal with a more complex situation, and do it programmatically in SAS.

Using SAS® to Parse External Data

Identifying Unique Character Strings

The human factor in parsing:

How do you separate your information into noise and useful data?

“Noise” will be rejected.

Useful data can be broken into 2 groups:

- data to be kept and subsequently processed
- identifiers that help to determine the difference between data and noise.

Using SAS® to Parse External Data Identifying Unique Character Strings

NOTE: There were 20000 observations read from the data set WORK1

NOTE: The PROCEDURE PRINT printed pages 1-364.

NOTE: The PROCEDURE PRINT used 0.24 CPU seconds and 7036K.

Noise

The address space has used a maximum of 620K below the line

NOTE: The SAS session used 0.24 CPU seconds and 7036K.

15 The SAS System 10:36 Sunday, January 20, 2008

Identifier

NOTE: The address space has used a maximum of 620K below the line

NOTE: SAS Institute Inc., SAS Campus Drive, Cary, NC USA 27513-

1 The SAS System 10:36 Sunday, January 20, 2008 1

Data

	Loop			
Obs	Index	Random1	Random2	
1	1	0.79940	90	
2	2	0.58974	53	
3	3	0.33958	79	
4	1	0.33424	37	
5	2	0.79739	3	

Using SAS® to Parse External Data

Input Statement – Named Input

Named Input:

```
INPUT Variable= ;
```

Has been a part of SAS since, well since I got started using it over 25 years ago!

Input data must contain data in format:

```
FieldName=value
```

Using SAS® to Parse External Data

Input Statement – Named Input

Named Input:

```
INPUT Variable= ;
```

This follows the same premise as the data parser we are discussing:

- Noise is ignored
- *FieldName* becomes the identifier
- *Value* is the useful data

Using SAS® to Parse External Data Input Statement – Named Input

EXAMPLE 2 : *Basic use of Named Input.*

```
DATA Example2;  
  INPUT  RANDOM3=;  
  DATALINES4;  
RANDOM3=94  
RANDOM3=334  
RANDOM3=966  
RANDOM3=809  
RANDOM3=934  
RANDOM3=889  
  
iiii  
proc print uniform;  
run;
```



The SAS System

Obs	RANDOM3
1	94
2	334
3	966
4	809
5	934
6	889

Using SAS® to Parse External Data Input Statement – Named Input

EXAMPLE 2b : *Basic use of Named Input.*

```
DATA Example2;  
    INPUT  RANDOM3= ;  
    DATALINES4;  
Sample Header Line  
1 RANDOM2=94 RANDOM3=94  
2 RANDOM2=334 RANDOM3=334  
3 RANDOM2=966 RANDOM3=966  
4 RANDOM2=809 RANDOM3=809  
5 RANDOM2=934 RANDOM3=934  
6 RANDOM2=889 RANDOM3=889  
iiii
```

Using SAS® to Parse External Data Input Statement – Named Input

EXAMPLE 2b : *Basic use of Named Input.*

```
47 DATA Example2;
48     INPUT  RANDOM3= ;
49 DATALINES4;
NOTE: EQUAL SIGN not found.
NOTE: NAME, '1 RANDOM2' is not defined.
RULE:      -----+-----1-----+-----2-----+-----
951          1 RANDOM2=94 RANDOM3=94
RANDOM3=94  _ERROR_=1  _N_=2
                ... ..
```

NOTE: The data set WORK.EXAMPLE2 has
7 observations and 1 variables.

Using SAS® to Parse External Data Input Statement – Named Input

EXAMPLE 2b : *Basic use of Named Input.*

```
DATA Example2;  
  INPUT  RANDOM3= ;  
  DATALINES4 ;  
sample Header Line  
1  RANDOM2=94  RANDOM3=94  
2  RANDOM2=334  RANDOM3=334  
3  RANDOM2=966  RANDOM3=966  
4  RANDOM2=809  RANDOM3=809  
5  RANDOM2=934  RANDOM3=934  
6  RANDOM2=889  RANDOM3=889  
; ; ; ;
```

Using SAS® to Parse External Data Input Statement – Named Input

EXAMPLE 2b: *Basic use of Named Input.*

```
DATA Example2 (KEEP=RANDOM3) ;  
OR RETAIN    RANDOM2    0 ;  
OR LENGTH   RANDOM2    8. ;  
INPUT  dummyval    $CHAR1 .  
      RANDOM3=    FIRSTOBS=2 ;  
OR IF RANDOM3 ^= . THEN OUTPUT ;  
  DATALINES4 ;  
  Sample Header Line  
  1 RANDOM2=94  RANDOM3=94  
  2 RANDOM2=334  RANDOM3=334  
  3 RANDOM2=966  RANDOM3=966  
  4 RANDOM2=809  RANDOM3=809  
  5 RANDOM2=934  RANDOM3=934  
  6 RANDOM2=889  RANDOM3=889
```

;;;;

(modified)

Using SAS® to Parse External Data Input Statement – Named Input

EXAMPLE 2b: *Basic use of Named Input.*

```
66 DATA Example2(KEEP=RANDOM3) ;
67     RETAIN    RANDOM2    0;
68     INPUT    dummyval $CHAR1.
69             RANDOM3= ;
70     IF    RANDOM3 ^= .    THEN OUTPUT;
71 DATALINES4 :
```

NOTE: EQUAL SIGN not found.

WHY?

NOTE: The data set WORK.EXAMPLE2 has 6 observations and 1 variables.

NOTE: DATA statement used (Total process time):

real time	0.00 seconds
cpu time	0.00 seconds

(modified)

Using SAS® to Parse External Data

Input Statement – Line and Column Pointers

Line and Column Pointers:

INPUT / *Variable* ;

INPUT # <num> *Variable* ;

Line Pointers

/ : **Relative – Jump one line down.**

Values not allowed; 1 slash = 1 line down.

: **Absolute - Specifies a line in the**

current

buffer, up or down. Limited by size

of

buffer. N= option on INFILE

Using SAS® to Parse External Data Input Statement – Line and Column Pointers

Line and Column Pointers:

```
INPUT / Variable ;
```

```
INPUT # <num> Variable ;
```

Line Pointers

~~Can only go forward – not backwards.~~

N= option on **INFILE** statement controls
the number of lines available.

The default is 1.

Using SAS® to Parse External Data Input Statement – Named Input

EXAMPLE 3 : *Absolute and Relative Column pointer.*

```
DATA TEMP;  
  RETAIN  pnt 2 ;  
  INFILE DATALINES4;  
  INPUT # 1  X  
        # pnt Y  
        # (pnt**2-1) Z ;
```



The SAS System

Obs	pnt	X	Y	Z
1	2	1	2	3
2	2	4	5	6

```
DATALINES;
```

```
1
```

```
2
```

```
3
```

```
4
```

```
5
```

```
6
```

```
;;;;
```

Using SAS® to Parse External Data Input Statement – Named Input

EXAMPLE 3 : *Absolute and Relative Column pointer.*

```
DATA TEMP;                                55 DATA TEMP;
  RETAIN pnt 3 ;                            56   RETAIN pnt 3 ;
  INFILE DATALINES4;                       57   INFILE DATALINES4;
  INPUT # 1 X                                58   INPUT # 1 X
        # pnt Y                               59         # pnt Y
        # (pnt-1) Z ;                         60         # (pnt-1) Z ;
DATALINES;                                  61 DATALINES;
1                                             ERROR: Old line 63 wanted but SAS
2                                             is at line 64.
3                                             Use: INFILE N=X; , with
4                                             a suitable value of x.
5                                             RULE:  -----+-----1-----+-----2-----+-
6                                             64      3
;;;                                         pnt=3 X=1 Y=3 Z=.  _ERROR_=1  _N_=1
```

Using SAS® to Parse External Data Input Statement – Named Input

EXAMPLE 3 : *Absolute and Relative Column pointer.*

```
DATA TEMP;                                RULE:  ----+----1----+----2----+-
      RETAIN  pnt 3 ;                      64    3
      INFILE DATALINES4;                  pnt=3 X=1 Y=3 Z=. _ERROR_=1 _N_=1
      INPUT # 1 X                          NOTE: The SAS System stopped
          # pnt Y                            processing this step
          # (pnt-1) Z ;                      because of errors.

DATALINES;                                WARNING: The data set WORK.TEMP
1                                           may be incomplete.  When
2                                           this step was stopped
3                                           there were 0 observations
4                                           and 4 variables.
5                                           WARNING: Data set WORK.TEMP was
6                                           not replaced because
      ; ; ; ;                              this step was stopped.
```

Using SAS® to Parse External Data Input Statement – Line and Column Pointers

Line and Column Pointers:

INPUT + <num> Variable ;

INPUT @ <pos> Variable ;

Column Pointers

+ : Relative – Move to left (or right).

**@ : Absolute - Specifies a line in the
current
buffer, up or down.**

Values, variables, and expressions are allowed.

Using SAS® to Parse External Data

Input Statement – Line and Column Pointers

Line and Column Pointers:

```
INPUT + (-NegativePos) Variable ;
```

Column Pointers

Relative – Can specify a negative number or expression – use parentheses.

Using SAS® to Parse External Data Input Statement – Line and Column Pointers

Line and Column Pointers:

```
INPUT @ "CharStr" Variable ;
```

Column Pointers

Absolute – Can specify a character string.

Column pointer will be moved 1 position to the right after the character string.

Using SAS® to Parse External Data Input Statement – Line and Column Pointers

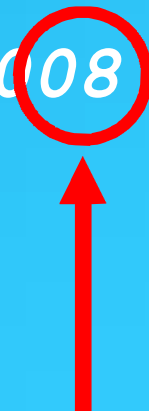
Line and Column Pointers:

```
INPUT @ "CharStr" Variable ;
```

Column Pointers

```
INPUT @ "20" TwoDigitYear 2.;
```

```
The current date is March 18 2008
```



Using SAS® to Parse External Data Input Statement – Line and Column Pointers

~~EXAMPLE 2b: Basic use of Named Input~~

EXAMPLE 4:

Using character-based column pointer.

```
DATA Example2;  
  INPUT @ "RANDOM3=" RANDOM3;  
  DATALINES4;
```

Sample Header Line

```
1 RANDOM2=94 RANDOM3=94  
2 RANDOM2=334 RANDOM3=334  
3 RANDOM2=966 RANDOM3=966  
4 RANDOM2=809 RANDOM3=809  
5 RANDOM2=934 RANDOM3=934  
6 RANDOM2=889 RANDOM3=889
```

iiii

*(modified)
differently*

Using SAS® to Parse External Data

Useful Character Functions

EXAMPLE 4:

Using character-based column pointer.

```
The SAS System                14:24 Thursday, January 17, 2008    1
  RANDOM6=20  RANDOM7=5564  RANDOM3=94  RANDOM2=8534  RANDOM4=69
  RANDOM3=334  RANDOM6=20  RANDOM8=2805  RANDOM1=8  RANDOM4=38
RANDOM2=5495  RANDOM8=8324  RANDOM6=72  RANDOM4=76  RANDOM3=966
  RANDOM5=8675  RANDOM2=6437  RANDOM1=6  RANDOM3=809  RANDOM6=68
RANDOM2=1753  RANDOM5=6451  RANDOM6=69  RANDOM4=93  RANDOM3=934
  RANDOM8=231  RANDOM3=889  RANDOM1=9  RANDOM4=55  RANDOM7=3448
  RANDOM4=12  RANDOM2=2910  RANDOM6=43  RANDOM3=312
  RANDOM7=8442  RANDOM6=26  RANDOM3=145  RANDOM2=3227  RANDOM1=3
  RANDOM2=5202  RANDOM1=3  RANDOM8=5330  RANDOM4=55  RANDOM6=24
RANDOM4=45  RANDOM1=4  RANDOM7=4885  RANDOM2=8639  RANDOM5=7358
RANDOM1=2  RANDOM2=6694  RANDOM6=6  RANDOM3=639  RANDOM4=85
  RANDOM5=5437  RANDOM3=836  RANDOM6=51  RANDOM7=3761  RANDOM1=5
```

Using SAS® to Parse External Data

Useful Character Functions

EXAMPLE 4b:

Using character-based column pointer.

```
DATA Example3;
  INFILE 'C:\HOW\Kuligowski\Example2.txt';
  INPUT ;
  IF FIND( _INFILE_, "RANDOM8=" ) THEN
    Result = 'success';
  ELSE
    Result = 'failure';
  PUT _ALL_ ;
RUN;
proc print uniform; run;
```

Using SAS® to Parse External Data

Useful Character Functions

Locate a set of characters in a larger set of characters:

```
INDEX( haystack, needle)
```

```
FIND( haystack, needle <,mods>, <startpos>)
```

How to remember? – “One of these things is not like the others”:

```
INDEX( haystack, needle)
```

Using SAS® to Parse External Data

Useful Character Functions

Cut the string back:

`SUBSTR(string, start-pos <, length>)`

`SCAN(string, word-number <, delimiters>)`

Check a string against another string:

`COMPARE(string1, string2)`

Using SAS® to Parse External Data

Useful Character Functions

Look for the presence of ... :

`ANYALNUM (string, <,start-pos>)`

`ANYALPHA (string, <,start-pos>)`

`ANYCNTRL (string, <,start-pos>)`

`ANYDIGIT (string, <,start-pos>)`

`ANYFIRST (string, <,start-pos>)`

`ANYGRAPH (string, <,start-pos>)`

`ANYLOWER (string, <,start-pos>)`

... and 6 other similar functions.

Using SAS® to Parse External Data

Looping & Conditional Execution

Quick list of conditional constructs:

`IF expression THEN / ELSE`

`SELECT <expression> / WHEN / OTHERWISE`

Quick list of looping constructs:

`DO WHILE expression`

`DO UNTIL expression`

To get out immediately, simply `LEAVE`

Using SAS® to Parse External Data

Putting it all together ...

Now the fun begins!

Pull out / dust off your toolkit.

You won't need everything you've collected to date.

On the other hand, you may have to research some new technique.

Burger Trough, Inc.
World Headquarters
Seven LittleFoy Road
San Antonio, TX 78205

Mr. Andrew P. Kuligowski
1234 Fakename St.
Dunedin, FL 34698

07 January 2008

Dear Mr. Kuligowski,

Thank you for your recent inquiries to the nutritional contents of our world-famous Troughburger Economy Meal. It may be innomdest, but we practically brag that we lead the industry in Total Fat AND Sodium, and are among the industry leaders in Cholesterol; our competitors come woefully short of the mark. Further, we consistently dominate our competitors by having MORE calories and LESS protein than the 3 companies with the highest market share - COMBINED!!

As per government regulations, we are happy to provide you with the details backing our claims. Please refer to the attached table.

Nutritional Facts

Amount per serving	Troughburger	Belgian Fries	CassidyCola
Calories:	1480	1140	620
Calories from Fat	760	540	0
Total Fat	84g 130%	60g 94%	0g 0%
Saturated Fat	38g 192%	12g 60%	0g 0%
Trans Fat	5g	16g	0g
Cholesterol	310mg 104%	15mg 5%	0mg 0%
Sodium	2760mg 114%	660mg 28%	40mg 2%
Total Carbohydrates	80g 39%	70g 46%	172g 58%
Dietary Fiber	6g 24%	14g 56%	0g 0%
Sugars	18g	0g	172g
Protein	12g	3g	0g

Thank you again for your interest and continued support of Burger Trough.

Sincerely,

Carl "Bo" Hyde
Sr. Vice President, Health & Quality Division

® to Parse External Data
Putting it all together ...

all together

Noise

Data

Noise

Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 5 : Putting it all together

Nutritional Facts

Amount per serving	Troughburger		Belgian Fries		Gas & Cola	
Calories	1480		1140		620	
Calories from Fat	760		540		0	
Total Fat	84g	120%	60g	94%	0g	0%
Saturated Fat	38g	192%	12g	60%	0g	0%
Trans Fat	5g		16g		0g	
Cholesterol	210mg	104%	15mg	5%	0mg	0%
Sodium	2760mg	114%	660mg	28%	40mg	2%
Total Carbohydrates	80g	29%	70g	46%	172g	58%
Dietary Fiber	6g	24%	14g	56%	0g	0%
Sugars	18g		0g		172g	
Protein	12g		3g		0g	

Thank you again for your interest and continued support of Burger Trough.

Identifier
Start of Data

Identifier
Body of Data

Identifier
End of Data

Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 5 : Putting it all together

```
DATA Example4;
    RETAIN  KeepRec_Ind 0;
    INFILE < filename / fileref >;
    INPUT  ;
*   Identify start of the nutritional table.  ;
*IF INDEX(_INFILE_,'Nutritional Facts') > 0 THEN DO;
*IF INDEX(UPCASE(_INFILE_),'Nutritional Facts') > 0 THEN DO;
    IF INDEX(UPCASE(_INFILE_),'NUTRITIONAL FACTS') > 0 THEN DO;
        PUTLOG 'Start of Nutritional Table identified' ;
        KeepRec_Ind = 1 ;
    END;
```

...

Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 5 : Putting it all together

```
...
* Identify body of the nutritional table.          ;
*IF INDEX(_INFILE_,'Amount per Serving') > 0 THEN DO;
*IF INDEX(UPCASE(_INFILE_), 'Amount per Serving') > 0 THEN DO;
  IF INDEX(UPCASE(_INFILE_), 'AMOUNT PER SERVING') > 0 THEN DO;
    PUTLOG 'Body of Nutritional Table identified' ;
    KeepRec_Ind = 2 ;
  END;
...

```

Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 5 : Putting it all together

```
...
* Identify termination of the nutritional table. ;
  IF KeepRec_Ind = 2 AND _INFILE_ = " " THEN DO;
    PUTLOG 'End of Nutritional Table identified' ;
  KeepRec_Ind = 0 ;
  END;
  IF KeepRec_Ind THEN DO;
    OutRec = _INFILE_ ;
  OUTPUT ;
  END ;
RUN ;
```

Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 5 : Putting it all together

```
Nutritional Facts
Amount per serving  Troughburger  Belgian Fries  Gas & Cola
Calories:           1480           1140           620
Calories from Fat   560           540           0
Total Fat           84g    120%    60g    94%    0g    0%
  Saturated Fat     38g    192%    12g    60%    0g    0%
  Trans Fat         5g
Cholesterol         210mg  104%    15mg    5%    0mg    0%
Sodium              2760mg 114%    660mg  28%    40mg   2%
Total Carbohydrates 80g    29%    70g    46%    172g  58%
  Dietary Fiber      6g    24%    14g    56%    0g    0%
  Sugars             18g
Protein             12g           3g           0g
```

Thank you again for your interest and continued support of Burger Trough.

**Identifier
Labels**

Label

**Label with
embedded
blank**

Label

Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 5 : Putting it all together

...

```
IF INDEX(UPCASE(_INFILE_), 'NUTRITIONAL FACTS') > 0 THEN DO;
  PUTLOG 'Start of Nutritional Table identified' ;
  KeepRec_Ind = 1 ;
END;
* Identify body of the nutritional table. ;
IF INDEX(UPCASE(_INFILE_), 'AMOUNT PER SERVING') > 0 THEN
DO;
  PUTLOG 'Body of Nutritional Table identified' ;
  KeepRec_Ind = 2 ;
END;
```

...

Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 5 : Putting it all together

...

```
ELSE IF INDEX(UPCASE(_INFILE_), 'AMOUNT PER SERVING') > 0
THEN DO;
    PUTLOG 'Body of Nutritional Table identified' ;
    KeepRec_Ind = 2 ;
    LengthStr = LENGTH( TRIM(_INFILE_) ) ;
    ScanPos = LENGTH('AMOUNT PER SERVING') ;
```

...

Toggle to the routine (Example 4b)

Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 5 : Putting it all together

...

```
DO WHILE( ScanPos < LengthStr);  
  ScanPos = ScanPos + 1 ;  
  NextChar = SUBSTR( _INFILE_, ScanPos, 1 );  
  IF NextChar = ' ' OR ScanPos = LengthStr THEN DO;  
    IF ScanPos = LengthStr THEN DO;  
      OneBlankInd = 1;  
      Product_Label = TRIM(Product_Label) || NextChar;  
    END;  
  END;
```

...

Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 5 : Putting it all together

...

```
IF OneBlankInd AND Product_Label ^= ' ' THEN DO;
  Product_Cnt + 1;
  OUTPUT;
  OneBlankInd = 0;
  Product_Label = ' ';
END;
ELSE OneBlankInd = 1;
END;
```

...

Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 5 : Putting it all together

...

```
ELSE DO;
  IF OneBlankInd THEN DO;
    Product_Label = TRIM(Product_Label) || ' ' ||
      NextChar ;
    OneBlankInd = 0;
  END;
  ELSE Product_Label = TRIM(Product_Label) ||
    NextChar ;
END;
END;
```

```
KeepRec_Ind = 3;
```

...

Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 5 : Putting it all together

Nutritional Facts

Amount per serving	Troughbasser		Belgian Fries		GasvCola	
Calories:	1480		1140		620	
Calories from Fat	760		540		0	
Total Fat	84g	120%	60g	94%	0g	0%
Saturated Fat	38g	192%	12g	60%	0g	0%
Trans Fat	5g		16g		0g	
Cholesterol	210mg	104%	15mg	5%	0mg	0%
Sodium	2760mg	114%	660mg	28%	40mg	2%
Total Carbohydrate	80g	29%	70g	46%	172g	58%
Dietary Fiber	6g	24%	14g	56%	0g	0%
Sugars	18g		0g		172g	
Protein	12g		3g		0g	

Thank you again for your interest and continued support of Burger Trough.

1 Column
vs.
2 Columns
Position of
Single
Column
Numbers

Additional Challenges

Units –
Yes or No?

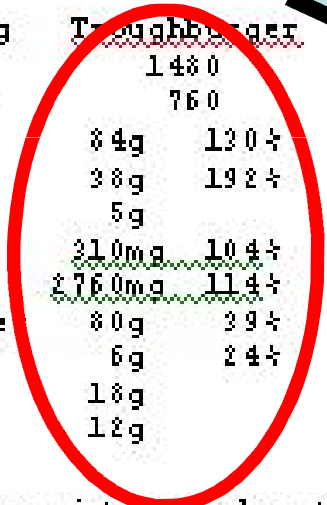
Using SAS® to Parse External Data

Putting it all together ...

EXAMPLE 4c: Putting it all together

Nutritional Facts

Amount per serving	Troughbasser		GasvCola	
Calories:	1480		620	
Calories from Fat	760		0	
Total Fat	84g	120%	60g	120%
Saturated Fat	38g	192%	12g	24%
Trans Fat	5g		16g	
Cholesterol	210mg	104%	15mg	5%
Sodium	2760mg	114%	660mg	28%
Total Carbohydrate	80g	29%	70g	46%
Dietary Fiber	6g	24%	14g	56%
Sugars	18g		0g	
Protein	12g		3g	



Toggle to the routine (Example 4c & 4d)

1 Column
vs.
2 Columns
Position of
Column
Numbers

Additional Challenges

Units –
Yes or No?

Using SAS® to Parse External Data Putting it all together ...

The next challenge(s) ...

- **Make the routine flexible – what if the letter contains a 4th product or more?**
- **Make the routine flexible – what if the letter spanned the page, and there was some additional header info at the top of the next page?**
- **Make the parser table driven.**

Many parsers can take a long time to code and test.

Parsers are only as good as the stability of the data they are reading. If you are sent an update – will *the data* still be in the same format?

Consider an alternative to writing a data parser:

- a) Ask the data supplier to send it in a more “friendly” format.**
- b) Cut-and-paste the data?**
- c) Manual data entry?**

Using SAS® to Parse External Data

The future ... is now!

SAS 9.2 and future releases may offer additional tools to assist parsing:

For example, from Version 9.2:

DLMSTR= 'xx' instead of DLM= 'x' will permit the use of multiple characters as a delimiter.

Using SAS® to Parse External Data

Conclusion

Make sure you have a good understanding of the input “format” (such as it is).

Any tool in your SAS arsenal may prove useful when writing a data parser.

INFILE & INPUT

IF / THEN / ELSE

SELECT / WHEN / OTHERWISE

DO UNTIL & DO WHILE

Functions – Character evaluation

Using SAS® to Parse External Data

Conclusion

BIG RED FLAG:

See if you can get the data in a cleaner format before undertaking the effort!

Using SAS® to Parse External Data Conclusion

Chances are, your “special format” will not look anything like any of the ones we discussed in this presentation.

However, the concepts of analysis and coding should be VERY similar!

Using SAS® to Parse External Data

Conclusion

Contact the author:

KuligowskiAndrew@gmail.com

AKuligowsk@fcc-group.com

Using SAS® to Parse External Data

Conclusion

SAS
is a registered trademark or trademark of SAS Institute, Inc. in the USA and other countries. (R) indicates USA registration.

*My
lawyer*

