CHARACTER DATA – Acquisition, Manipulation, and Analysis

Andrew T. Kuligowski, HSN
Swati Agarwal, Optum

SAS GLOBAL FORUM

CHARACTER DATA – Acquisition, Manipulation, and Analysis

Andrew T. Kuligowski, HSN
Swati Agarwal, Optuminsight

SAS GLOBAL FORUM

Agenda

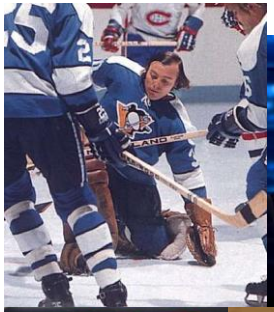
What is “Character Data”?

How long is my Character Variable?

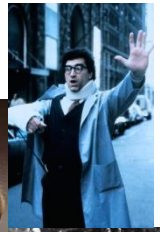
Manipulating Character Data

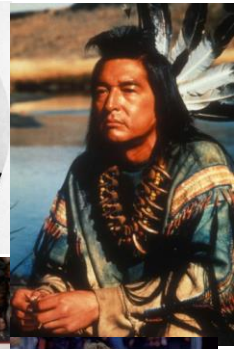
- **Truncation Functions**
- **Concatenation Functions**
- **Length Functions**
- **Change Case Functions**
- **Substring Functions**
- **Misc. Functions**

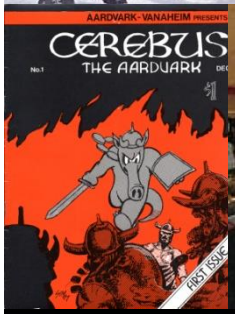
“Characters”?




NBC
NIGHTLY
NEWS
WITH BRIAN WILLIAMS






AARDVARK-VANAHEIM PRESENTS
CEREBUS
THE AARDVARK


GROSSMANN
8


00

What is “Character Data”?

“*Character data consists of any combination of letters, symbols, and numeric characters.*”

Quote from Microsoft documentation

ASCII

EBCDIC

One byte per letter (upper and lower case), or number, or select special characters (the ones on a standard typewriter), plus a few non-printable: Carriage Return, Line Feed, Tab, Bell, etc.

SAS GLOBAL FORUM

What is “Character Data”?

“*Character data consists of any combination of letters, symbols, and numeric characters.*”

Quote from Microsoft documentation

ASCII

EBCDIC

What about alphabetic characters used in other languages? Æ Ç Ñ

What about special punctuation and symbols? ™ ¿

What about curly quotes? “ ”

SAS GLOBAL FORUM

What is “Character Data”?

“*Character data consists of any combination of letters, symbols, and numeric characters.*”

Quote from Microsoft documentation

Unicode

Many more characters can be defined and stored.

BUT … it takes more than one byte of storage to do so. (Nothing is free ...)

SAS GLOBAL FORUM

How Long is a Character Variable?

The default length of a character is 200 characters (unless, of course, overridden).

SAS GLOBAL FORUM

How Long is a Character Variable?

The default length of a character is 200 characters (unless, of course, overridden).

NOT ANYMORE

The MAXIMUM length of a character variable also used to be 200 bytes – now it's 32,767 bytes.
(*THAT* would be an undesirable default.)

SAS GLOBAL FORUM

How Long is a Character Variable?

The default length of a character is now based on its first use. (Again, unless overridden.)

Hardcoded Value

The length of the FIRST value to which the variable is set (which may not be the longest value … well, for better or worse, it just became the longest value.)

How Long is a Character Variable?

The default length of a character is now based on its first use. (Again, unless overridden.)

Value built from other character variable(s)

The TOTAL length of the variable(s) used to build the new variable.

If you want to be safe, use a LENGTH statement – at the *top* of your DATA step.

SAS GLOBAL FORUM

How Long is a Character Variable?

```
data temp;
length w $ 2  y $ 40.;
retain w "";
x = "abcdefghijklmnopqrstuvwxyz";
y = "";
z = "1234567890";
output;
Y = "----+----1----+----2----+" ||
    "----3----+----4----+----5";
a1 = substr(x,14);
a2 = substr(x,14,13);
a3 = x || y ;
output;
run;
```

Alphabetic List of Variables and Attributes

#	Variable	Type	Len
5	a1	Char	26
6	a2	Char	26
7	a3	Char	66
1	w	Char	2
3	x	Char	26
2	y	Char	40
4	z	Char	10

How Long is a Character Variable?

You must define LENGTH before the first reference to the variable in your code. Otherwise ...

WARNING: Length of character variable y has already been set. Use the LENGTH statement as the very first statement in the DATA STEP to declare the length of a character variable.

```
da    temp;
le      $ 2   y $
r
a1 = sub
a2 = substr(
a3 = x || y ;
output;
run;
```

List of Variables and Attributes

Variable	Type	Len
a1	Char	26
a2	Char	26
a3	Char	66
w	Char	2
x	Char	26
y	Char	40
z	Char	10

How Long is a Character Variable?

Different lengths when 2 or more datasets are brought together?

Longer one referenced first – <silence>

Shorter one referenced first – WARNING: Multiple lengths were specified for the variable String by input data set(s). This may cause truncation of data.

```
data te
length
r
a1 =
a2 = subs
a3 = x || y ;
output;
run;
```

Alphabetic List of Variables and Attributes

#	Variable	Type	Len
5	a1	Char	26
6	a2	Char	26
3	a	Char	66
1	v	Char	2
3	x	Char	26
2	y	Char	40
4	z	Char	10

Manipulation: Truncation Functions

- LEFT()
- RIGHT()
- TRIM()
- TRIMN()
- STRIP()
- COMPRESS()
- COMPBL()
- REPEAT()
- REVERSE()

SAS GLOBAL FORUM

LEFT() vs. RIGHT()

- **Functions designed to aligned character variables:**
- **LEFT( )**
- **RIGHT( )**

No CENTERED() - you'll have to code that for yourselves.

LEFT() vs. RIGHT()

```
data sample;
  set sample;
  left = '*' || left(string) || '*';
  right = '*' || right(string)   || '*';
run;
```

The SAS System

Obs	string	Left			right
1	Joe Smith	*Joe Smith	*	*	Joe Smith*
2	Roma Brown	*Roma Brown	*	*	Roma Brown*
3	Alice Wonde	*Alice Wonde	*	*	Alice Wonde*
4	Li Wang	*Li Wang	*	*	Li Wang*
5		*	*	*	*

Note alignment in each value

TRIM() vs. TRIMN()

- Functions designed to truncate character variables:
- TRIM( )
- TRIMN( )
- STRIP( )
- COMPRESS( )

SAS GLOBAL FORUM

TRIM() vs. TRIMN()

```
data sample;
  input string $char14.;
datalines;
Joe Smith
   Roma Brown
   Alice Wonderland
   Li    Wang

;
```

contains trailing blanks
contains leading blanks
contains leading and trailing blanks
contains leading, trailing blanks and multiple blanks in between
(last line contains a blank string)

TRIM() vs. TRIMN()

```
data sample;
  input string $char1.;
datalines;
Joe Smith
   Roma Brown
   Alice Wonderland
   Li    Wang

;
```

```
data sample;
set sample;
original = '*' || string || '*';
trim = '*' || trim(string) || '*';
trimn = '*' || trimn(string) || '*';
run;
```

SAS GLOBAL FORUM

TRIM() vs. TRIMN()

The SAS System
Output of TRIM and TRIMN

original	trim	trimn
*Joe Smith *	*Joe Smith*	*Joe Smith*
* Roma Brown *	* Roma Brown*	* Roma Brown*
* Alice Wonde*	* Alice Wonde*	* Alice Wonde*
* Li Wang *	* Li Wang*	* Li Wang*
* *	* *	**

Note difference in how blank value is handled.
Note lack of difference in how other values are handled

SAS GLOBAL FORUM

STRIP() vs. TRIM() vs. TRIMN()

- Functions designed to truncate character variables:
- TRIM( )
- TRIMN( )
- STRIP( )
- COMPRESS( )

STRIP( ) vs. TRIM(LEFT( ))
vs. TRIMN(LEFT( ))

Each will remove trailing blanks.

STRIP( ) will also remove leading blanks.

SAS GLOBAL FORUM

STRIP() vs. TRIM() vs. TRIMN()

```
data sample;
set sample;
   strip       = '*' || strip(string)         || '*';
   trim_left   = '*' || trim(left(string))    || '*';
   trimn_left  = '*' || trimn(left(string))   || '*';
run;
```

The SAS System
Output of STRIP, TRIM(LEFT) and TRIMN(LEFT)

original	strip	trim_left	trimn_left
*Joe Smith *	*Joe Smith*	*Joe Smith*	*Joe Smith*
* Roma Brown *	*Roma Brown*	*Roma Brown*	*Roma Brown*
* Alice Wonde*	*Alice Wonde*	*Alice Wonde*	*Alice Wonde*
* Li Wang *	*Li Wang*	*Li Wang*	*Li Wang*
* *	**	* *	**

No difference between STRIP and TRIMN(LEFT( )) (except *efficiency*)

COMPRESS() vs. COMPBL()

- Functions designed to truncate character variables:
- TRIM( )
- TRIMN( )
- STRIP( )
- COMPRESS( )

COMPRESS removes a specified character(s) from a string.
Default = “ ” if no characters specified.

COMPBL allows compression of multiple blanks into a single blank

COMPRESS() vs. COMPBL()

```
data sample;
   set sample;
   compress = '*' || compress(string) || '*';
   compbl   = '*' || compbl(string)   || '*';
run;
```

Output of COMPRESS and COMPBL

original	compress	compbl
*Joe Smith *	*JoeSmith*	*Joe Smith *
* Roma Brown *	*RomaBrown*	* Roma Brown *
* Alice Wonde*	*AliceWonde*	* Alice Wonde*
* Li Wang *	*LiWang*	* Li Wang *
* *	**	* *

Note number and location of blanks in each value – including leading blanks.

COMPRESS() vs. COMPBL()

```
data zipcode;
input zipcode $14.;
zipcode1 = compress(zipcode);
zipcode2 = compress(zipcode,'  ()?');
zipcode3 = compress(zipcode,' - ()?');
zipcode4 = compress(zipcode,'ABCD','A');
zipcode5 = compress(zipcode,'23456','k');
datalines;
     22168- 12 34
     22168- (1234?)
     MN55346 - mn44
;
```

1 – removes blanks
2 – remove blanks, parens, question mark
3 – remove blanks, parens, question marks, and dashes.
4 – REMOVE specified characters ABCD
5 – KEEP specified characters 23456

COMPRESS() vs. COMPBL()

```
data zipcode;
input zipcode $14.;
zipcode1 = compress(zipcode);
zipcode2 = compress(zipcode, '()?');
zipcode3 = compress(zipcode, '()?-');
zipcode4 = compress(zipcode, 'ABCD', 'A');
```

1 – removes blanks
2 – remove blanks, parens, question mark
3 – remove blanks, parens, question marks, and dashes.
4 – REMOVE specified characters ABCD
5 – KEEP specified characters 23456

The SAS System
Listing of Zipcodes

zipcode	zipcode1	zipcode2	zipcode3	zipcode4	zipcode5
22168- 12 34	22168-1234	22168-1234	221681234	22168- 12 34	226234
22168- (1234?)	22168-(1234?)	22168-1234	221681234	22168- (1234?)	226234
MN55346 - mn44	MN55346-mn44	MN55346-mn44	MN55346mn44	55346 - 44	5534644

REPEAT() vs. REVERSE()

- **Functions designed to aligned character variables:**
- **REVERSE( )** Reverses SAS character expression, by default the variable length is determined by the length of the first argument.
- **REPEAT( )** Repeats the Character expression, by default the variable length is of 200 bytes.

REPEAT() vs. REVERSE()

```
data sample;
   set sample;
   backward = reverse(string);
   repeat   = repeat(string,1);
run;
```

The SAS System

Obs	string	backward	repeat
1	Joe	eoJ	Joe Joe
2	Roma Brown	nworB amoR	Roma BrownRoma Brown
3	Alice	ecilA	Alice Alice

Alphabetic List of Variables and Attributes

#	Variable	Type	Len
2	backward	Char	10
3	repeat	Char	200
1	string	Char	10

SAS GLOBAL FORUM

REPEAT() vs. REVERSE()

```
data sample;
   set sample;
   backward = reverse(string);
   repeat   = repeat(string,1);
run;
```

The SAS System

Obs	string	backward	repeat
1	Joe	eoJ	Joe Joe
2	Roma Brown	nworB amoR	Roma BrownRoma Brown
3	Alice	ecilA	Alice Alice

Alphabetic List of Variables and Attributes

#	Variable	Type	Len
2	backward	Char	10
3	repeat	Char	200
1	string	Char	10

SAS GLOBAL FORUM

Concatenation Functions

- CAT()
- CATT()
- CALL CATT Routine
- CATS()
- CALL CATS Routine
- CATX()
- CALL CATX Routine

SAS GLOBAL FORUM

Concatenation Functions

- Functions designed to concatenate character variables:

	Concatenates character strings …
CAT( )	… without removing leading or trailing blanks.
CATT( )	… and removes trailing blanks.
CATS( )	… and removes leading and trailing blanks.
CATX( )	… and removes leading and trailing blanks, and inserts separators between each string

Concatenation Functions

```
data sample;
  set sample;
  length cat catt cats $16 catx $20;
  text='Hello';
  cat =cat ('*',string,'*');
  catt=catt('*',string,'*');
  cats=cats('*',string,'*');
  catx=catx('!',text,string);
run;
```

= ||

= TRIM || or TRIMN ||

= STRIP ||

= STRIP || separator

SAS GLOBAL FORUM

Concatenation Functions

```
data sample;
   set sample;
   length cat catt cats $16 catx $20;
   text='Hello';
```

= ||

or TRIMN ||

||

|| separator

The SAS System
Output of Concatenation Functions

cat	catt	cats	catx
*Joe Smith *	*Joe Smith*	*Joe Smith*	Hello!Joe Smith
* Roma Brown *	* Roma Brown*	*Roma Brown*	Hello!Roma Brown
* Alice Wonde*	* Alice Wonde*	*Alice Wonde*	Hello!Alice Wonde
* Li Wang *	* Li Wang*	*Li Wang*	Hello!Li Wang
* *	**	**	Hello

SAS GLOBAL FORUM

Concatenation Functions

```
data sample;
  set sample;
  length cat catt cats $16 catx $20;
  text='Hello';
  cat =cat ('*',string,'*');
  catt=catt('*',string,'*');
```

CAT functions are more efficient than the combination of TRIM( ) and || for concatenation.

NOTE: Default Length is 200.

The SAS System

Output of Concatenation Functions

cat	catt	cats	catx
*Joe Smith *	*Joe Smith*	*Joe Smith*	Hello!Joe Smith
* Roma Brown *	* Roma Brown*	*Roma Brown*	Hello!Roma Brown
* Alice Wonde*	* Alice Wonde*	*Alice Wonde*	Hello!Alice Wonde
* Li Wang *	* Li Wang*	*Li Wang*	Hello!Li Wang
* *	**	**	Hello

SAS GLOBAL FORUM

Concatenation Functions

CALL CATT ROUTINE: The CALL CATT routine returns the result in the first argument, result.

```
CALL CATT(result <, string-1, ...string-n>);
```

CALL CATS ROUTINE: The CALL CATS routine returns the result in the first argument, result.

```
CALL CATS(result <, string-1, ...string-n>);
```

CALL CATX ROUTINE: The CALL CATX routine returns the result in the second argument, result.

```
CALL CATX(separator, result<, string-1 , ...>string-n);
```

Concatenation Functions

```
data _null_;
   length catt $ 40 cats $ 40 catx $ 50;
   x='Rio is t  ';
   y=' he Olym   ';
   z=' pic site for 2016. ';
   separator='&';
   call catt(catt,x,y,z);   put catt;
   call cats(cats,x,y,z);   put cats;
   catx=catx(separator,catx,x,y,z);   put catx;
run;
```

```
Rio is t he Olym pic site for 2016.
Rio is the Olympic site for 2016.
Rio is t&he Olym&pic site for 2016.
```

SAS GLOBAL FORUM

Manipulation : Length Functions

- **LENGTH()**
- **LENGTHN()**
- **LENGTHC()**
- **LENGTHM()**

SAS GLOBAL FORUM

Manipulation : Length Functions

- **Functions designed to report variable value length:**

	Returns the length of a character string …
LENGTH( )	… excluding trailing blanks, and returns 1 for a blank string.
LENGTHN( )	… excluding trailing blanks, and returns 0 for a blank string.
LENGTHC( )	… including trailing blanks.
LENGTHM( )	Returns the amount of memory (in bytes) that is allocated for a character string.

Manipulation : Length Functions

```
data how_long;
  one = 'SASGF2014 ';
  Two = ' ';
  length_one  = length(one);
  lengthn_one = lengthn(one);
  lengthc_one = lengthc(one);
  lengthm_one = lengthm(one);
  length_two  = length(two);
  lengthn_two = lengthn(two);
  lengthc_two = lengthc(two);
  lengthm_two = lengthm(two);
run;
```

Nonblank value with trailing blank.
Missing value (Blank character).

SAS GLOBAL FORUM

Manipulation : Length Functions

```
data how_long;
  one = 'SASGF2014 ';
  Two = ' ';
  length_one  = length(one);
  lengthn_one = lengthn(one);
  lengthc_one = lengthc(one);
  lengthm_one = lengthm(one);
  length two  = length(two);
  l
  l
  l
run
```

Nonblank value with trailing blank.
Missing value (Blank character).

The SAS System
Length of Variable

one	two	length_ one	lengthn_ one	lengthc_ one	lengthm_ one	length_ two	lengthn_ two	lengthc_ two	lengthm_ two
SASGF2014		9	9	10	10	1	0	1	1

Manipulation : Change case Functions

- **UPCASE()**
- **LOWCASE()**
- **PROPCASE()**

SAS GLOBAL FORUM

Manipulation : Change case Functions

- Functions designed to change character values:

UPCASE( )	converts all uppercase letters to lowercase letters.
LOWCASE( )	converts all lowercase letters to uppercase letters.
PROPCASE( )	capitalizes the first letter of word in a string, leaving – or converting – the others to lower case.

SAS GLOBAL FORUM

Manipulation : Change case Functions

```
data ds_1;
length Name $ 80. ;
input Name $ & ;
datalines;
rosy, mike and others
ROSY, MIKE AND OTHERS
ROSY, MIke aNd OTherS
;
```

```
data ds_2;
set ds_1;
/*convert it to other cases*/
upcase_var = upcase(Name);
propercase_var = propcase(Name);
lowcase_var =
   lowcase(upcase_var);
run;
```

SAS GLOBAL FORUM

Manipulation : Change case Functions

```
data ds_1;
length Name $ 80. ;
input Name $ & ;
datalines;
rosy, mike and others
ROSY, MIKE AND OTHERS
R
;
```

```
data ds_2;
set ds_1;
/*convert it to other cases*/
upcase_var = upcase(Name);
propercase_var = propcase(Name);
lowcase_var =
```

The SAS System
Listing of Names

Name	upcase_var	propercase_var	lowcase_var
rosy, mike and others	ROSY, MIKE AND OTHERS	Rosy, Mike And Others	rosy, mike and others
ROSY, MIKE AND OTHERS	ROSY, MIKE AND OTHERS	Rosy, Mike And Others	rosy, mike and others
ROSY, MIke aNd OTherS	ROSY, MIKE AND OTHERS	Rosy, Mike And Others	rosy, mike and others

Manipulation : Substring Functions

- **SUBPAD Function**
- **SUBSTR (left of =) Function**
- **SUBSTR (right of =) Function**
- **SUBSTRN()**

Manipulation : Substring Functions

SUBPAD( ) – will return a variable with the length specified, padding the results with spaces.

SUBSTR(left of =) – Replaces Character value contents

SUBSTR(right of =) – Extracts the substring

SUBSTRN( ) – will return a substring and allows with length of zero

SAS GLOBAL FORUM

Manipulation : Substring Functions

```
3529 DATA Substr_example;
3530 RETAIN Example "abcdefghijklmnopqrstuvwxyz";
3531 RETAIN Shortway "abcdefghijklmnopqrstuvwxyz";
3532 Longway = SUBSTR( Example, 1, 10 ) ||
3533            "123456" ||
3534            SUBSTR( Example, 17 ) ;
3535 SUBSTR( Shortway, 11, 6 ) = "123456" ;
3536 PUTLOG _ALL_;
3537 RUN;
Example =abcdefghijklmnopqrstuvwxyz
Shortway=abcdefghij123456qrstuvwxyz
Longway =abcdefghij123456qrstuvwxyz _ERROR_=0 _N_=1
```

SAS GLOBAL FORUM

Manipulation : Substring Functions

- **CHAR Function**
- **`CHAR( string, position )` is equivalent to `SUBSTR( string, position, 1 )`**
- **FIRST Function**
  `FIRST( string )` is equivalent to `SUBSTR( string, 1, 1 )`

SAS GLOBAL FORUM

Working with Character Data

ONLY QUICK OVERVIEW DUE TO TIME CONSTRAINTS.

MANY functions help find a character(s) within a character string:

INDEX( )

FIND( )

Search a character string for the presence of a specified string, return the 1st position of the latter within the former. (Search for “needle” in “haystack”.)

FIND searches for entire “needle”.

FINDC searches for any character in “needle” that exists in “haystack”.

FINDW searches for entire “needle” BUT it must be a “word” separated by a delimiter(s).

SAS GLOBAL FORUM

Working with Character Data

MANY functions help find a character(s) within a character string:

INDEX( ) Searches for a character string within a specified character string, returning the position of the first character in the former.

FIND( ) Searches for “needle” in “haystack”.)

FIND searches for any character “needle” that exists in “haystack”.

FINDC searches for any character “needle” that exists in “haystack”.

FINDW searches for entire “needle” BUT it must be a “word” separated by a delimiter(s).

ONLY QUICK OVERVIEW DUE TO TIME CONSTRAINTS.

SO WHAT IS THE DIFFERENCE?

FIND has allows options such as “STARTING POSITION” and various modifiers.

(In my opinion), FIND is easier to remember!

Working with Character Data

ONLY QUICK OVERVIEW DUE TO TIME CONSTRAINTS.

MANY functions help find a character(s) within a character string:

INDEX( )	Search a character string for the presence of a character NOT in a specified string, return the 1st position of the latter within the former. (Search for "hay" in "haystack", report back when you find something that is not "hay".)
FIND( )	
VERIFY( )	

SAS GLOBAL FORUM

Working with Character Data

ONLY QUICK OVERVIEW DUE TO TIME CONSTRAINTS.

MANY functions help find a character(s) within a character string:

ANYALNUM( )	**ANYGRAPH( )**	**ANYPUNCT( )**
ANYALPHA( )	**ANYLOWER( )**	**ANYSPACE( )**
ANYCNTRL( )	**ANYNAME( )**	**ANYUPPER( )**
ANYDIGIT( )	**ANYPRINT( )**	**ANYXDIGIT( )**
ANYFIRST( )		

Search a character string for <*something*>, return the 1st position of the latter within the former.

SAS GLOBAL FORUM

Working with Character Data

ONLY QUICK OVERVIEW DUE TO TIME CONSTRAINTS.

MANY functions help find a character(s) within a character string:

NOTALNUM( )	**NOTGRAPH( )**	**NOTPUNCT( )**
NOTALPHA( )	**NOTLOWER( )**	**NOTSPACE( )**
NOTCNTRL( )	**NOTNAME( )**	**NOTUPPER( )**
NOTDIGIT( )	**NOTPRINT( )**	**NOTXDIGIT( )**
NOTFIRST( )		

Search a character string for <something>, return the 1st position of something that IS *NOT* the latter within the former.

Manipulation : Misc. Functions

COUNT() - counts substrings of characters in a character string.

COUNTC() - counts individual characters in a character string

IFC() - Returns a character value of an expression based on whether the expression is true, false, or missing

IFN() - Returns a numeric value of an expression based on whether the expression is true, false, or missing

SAS GLOBAL FORUM

Manipulation : Misc. Functions

```
data temp;
xyz='This is a thistle? Yes, this is a thistle.';
howmanythis=count(xyz,'this');
howmanythis1=count(xyz,'this','i');
howmanyi = countc(xyz,'i');
run;
```

The SAS System 13:35 Monday, Septem

xyz	howmanythis	howmanythis1	howmanyi
This is a thistle? Yes, this is a thistle.	3	4	6

Manipulation : Misc. Functions

```
data grade;
input name $ grade;
performance = ifc(grade>80, 'Pass', 'Needs Improvement');
if grade>80 then perf='Pass                    ';
      else perf = 'Needs Improvement';
datalines;
John 74
Kareem 89
Kati 100
Maria 92
;
run;
```

The SAS System 13:35

Obs	name	grade	performance	perf
1	John	74	Needs Improvement	Needs Improvement
2	Kareem	89	Pass	Pass
3	Kati	100	Pass	Pass
4	Maria	92	Pass	Pass

Manipulation : Misc. Functions

```
data sales;
input TotSales;
comm=ifn(TotSales > 10000, TotSales*.05, TotSales*.02);
if TotSales > 10000 then do comm_td = TotSales*.05;end;
   else do;comm_td = TotSales*.02;end;
datalines;
25000
10000
500
10300
;
run;
```

The SAS System

Obs	Tot Sales	comm	comm_td
1	25000	1250	1250
2	10000	200	200
3	500	10	10
4	10300	515	515

Manipulation : Misc. Functions

MISSING() – If numeric and character expression contains missing value SAS returns ‘1’ else SAS returns ‘0’

TRANSLATE() – Replaces specific character

TRANWRD() - Replaces or removes all occurrences of a word in a character string

SAS GLOBAL FORUM

Manipulation : Misc. Functions

```
data name;
input @1  sal    $5. @6 fname $6.
      @12 lname $20.;
datalines;
Miss  Joan  Smith
Ms    Ann
Miss Alice Cooper
;
run;
```

```
data show;
set name;
Sal_tranwrd =
   tranwrd(sal, "Miss", "Ms");
Sal_translate =
   translate(sal_tranwrd, "MS", "Ms");
if missing(lname) then
   comments = 'Last name is missing';
run;
```

SAS GLOBAL FORUM

Manipulation : Misc. Functions

```
data name;
input
datal
Miss
Ms
Miss
;
run;
```

```
                                The SAS System                    13:35 Monday,

                                                  Sal_       Sal_
Obs    sal     fname    lname     tranwrd    translate          comments

 1     Miss    Joan     Smith      Ms          MS
 2     Ms      Ann                 Ms          MS        Last name is missing
 3     Miss    Alice    Cooper     Ms          MS
```

```
   Sal_tranwrd =
      tranwrd(sal, "Miss", "Ms");
   Sal_translate =
      translate(sal_tranwrd, "MS", "Ms");
   if missing(lname) then
      comments = 'Last name is missing';
   run;
```

SAS GLOBAL FORUM

Manipulation : Misc. Functions

```
data name;
input
datal
Miss
Ms
Miss
;
run;
```

```
                                The SAS System                    13:35 Monday,

                                                  Sal_       Sal_
Obs    sal     fname    lname     tranwrd    translate          comments

  1    Miss    Joan     Smith     Ms         MS
  2    Ms      Ann                Ms         MS          Last name is missing
  3    Miss    Alice    Cooper    Ms         MS
```

```
   Sal_tranwrd =
      tranwrd(sal, "Miss", "Ms");
   Sal_translate =
      translate(sal_tranwrd, "MS", "Ms");
   if missing(lname) then
      comments = 'Last name is missing';
   run;
```

Working with Character Data

Perl Regular Expressions

ONLY QUICK OVERVIEW DUE TO TIME CONSTRAINTS.

Introduced in Version 9

More complex than “traditional” SAS functions.

More robust than “traditional” SAS functions.

Example: Search for a 3-character string where the first two characters are numbers and the 3rd is a 0.

```
IF _N_ = 1 THEN
   PATTERN_NEEDLE = PRXPARSE( "/\d\d0/" );

PRXMATCH( PATTERN_NEEDLE, STRING_HAYSTACK )
```

Working with Character Data

Informats and Formats

ONLY QUICK OVERVIEW DUE TO TIME CONSTRAINTS.

$<*nn*>. vs.
$CHAR<*nn*>.

PROC FORMAT

PUT( ) function and
INPUT( ) function

SAS GLOBAL FORUM

Working with Character Data

Updates in recent releases

The K Functions are here!

KCOMPARE	KCOMPRESS	KCOUNT	KCVT
KINDEX	KINDEXC	KLEFT	KLENGTH
KLOWCASE	KREVERSE	KRIGHT	KSCAN
KSTRCAT	KSUBSTR	KSUBSTRB	
KTRANSLATE		KTRIM	KTRUNCATE
KUPCASE	KUPDATE	KUPDATEB	KVERIFY

SAS GLOBAL FORUM

Working with Character Data

Updates in recent releases

Version 9.3: `&SYSSIZEOFUNICODE.`

Version 9.4: `PUTC( )` and `PUTN( )` now allow justification.

Version 9.4: `SCAN( )` defaults resulting variable to length of 1st string processed.

Version 9.4: [NEW] `TYPEOF( )` indicates whether an argument is character or numeric. (Only for WHERE clauses and Graphic Template Language [GTL].)

SAS GLOBAL FORUM

Working with Character Data

Updates in recent releases

Version 9.4: [NEW] `FCOPY( )` copies a record from one FILEREF to another.

Version 9.4: [NEW] `DOSUBL( )` trades values between SAS and the CALLING ENVIRONMENT's macro variables (*not SAS macros*!)

Version 9.4: [NEW] `SHA256( )` returns the result of a message digest of a specified string. (Cryptologists, take note!!)

SAS GLOBAL FORUM

Working with Character Data

This was an OVERVIEW of just 35 functions – designed to stimulate the curiosity, not to teach everything known about character data and how to work with it.
We hope it has done that – have fun as you continue to learn and to grow.

Contact:
swati_agarwal@optum.com
KuligowskiAndrew@gmail.com

SAS GLOBAL FORUM

Thank you for your interest in this topic!