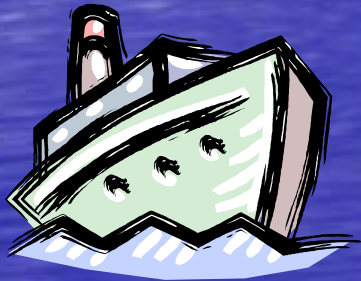


Character Function Fun

A voyage through character
functions added in SAS 9

by Tim Muir



The list

- ANYALNUM
- ANYALPHA
- ANYCNTRL
- ANYDIGIT
- ANYFIRST
- ANYGRAPH
- ANYLOWER
- ANYNAME
- ANYPRINT
- ANYPUNCT
- ANYSPACE
- ANYUPPER
- ANYXDIGIT
- CAT
- CATS
- CATT
- CATX
- CHOOSEC
- CHOOSEN
- COMPARE *
- COMPGED *
- COMPLEV *
- COUNT
- COUNTC
- FIND
- FINDC
- IFC
- IFN
- LENGTHC
- LENGTHM
- LENGTHN
- NLITERAL *
- NOTALNUM
- NOTALPHA
- NOTCNTRL
- NOTDIGIT
- NOTFIRST
- NOTGRAPH
- NOTLOWER
- NOTNAME
- NOTPRINT
- NOTPUNCT
- NOTSPACE
- NOTUPPER
- NOTXDIGIT
- NVALID *
- PROPCASE
- PRXCHANGE *
- PRXPOSN *
- SCANQ
- STRIP
- SUBPAD
- SUBSTRN

* maybe next time



ANYxxxx/ NOTxxxx functions

(26 index functions)

- **ANYALNUM (string, start)**
- **ANYALPHA (string, start)**
- **ANYCNTRL (string, start)**
- **ANYDIGIT (string, start)**
- **ANYFIRST (string, start)**
- **ANYGRAPH (string, start)**
- **ANYLOWER (string, start)**
- **ANYNAME (string, start)**
- **ANYPRINT (string, start)**
- **ANYPUNCT (string, start)**
- **ANYSpace (string, start)**
- **ANYUPPER (string, start)**
- **ANYXDIGIT (string, start)**
- **NOTALNUM (string, start)**
- **NOTALPHA (string, start)**
- **NOTCNTRL (string, start)**
- **NOTDIGIT (string, start)**
- **NOTFIRST (string, start)**
- **NOTGRAPH (string, start)**
- **NOTLOWER (string, start)**
- **NOTNAME (string, start)**
- **NOTPRINT (string, start)**
- **NOTPUNCT (string, start)**
- **NOTSPACE (string, start)**
- **NOTUPPER (string, start)**
- **NOTXDIGIT (string, start)**

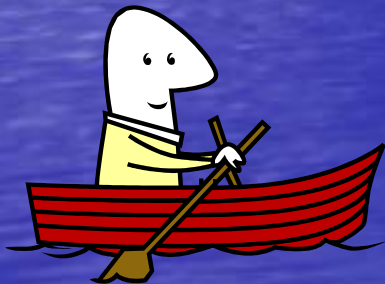
- start is optional
- If the value of start is positive, the search proceeds to the right.
- If the value of start is negative, the search proceeds to the left.
- If the value of start is less than the negative length of the string, the search begins at the end of the string.
- The function will return a 0 when:
 - the character that you are searching for is not found
 - the value of start is greater than the length of the string
 - the value of start = 0.

Contrast most these to indexc

- **ANYALNUM (variable)**
 - beats using:
`indexc(lowercase(variable),`qwertyuiopasdfghjklzxcvbnm1234567890`)`
- **ANYALPHA (variable)**
 - beats using: `indexc(lowercase(variable),`qwertyuiopasdfghjklzxcvbnm`)`
- **NOTALNUM (variable)**
 - beats using:
`indexc(lowercase(variable),`qwertyuiopasdfghjklzxcvbnm1234567890`)=0`
- **Etc.**

What's Left

- CAT
- CATS
- CATT
- CATX
- CHOOSEC
- CHOOSEN
- COUNT
- COUNTC
- FIND
- FINDC
- IFC
- IFN
- LENGTHC
- LENGTHM
- LENGTHN
- PROPCASE
- SCANQ
- STRIP
- SUBPAD
- SUBSTRN



cat/cats/catt/catx

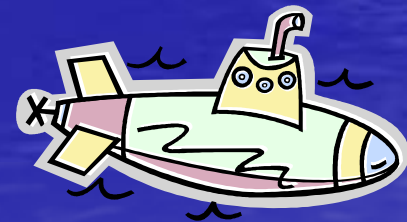
(concatenation functions)

- Differences:
 - CAT: concatenation without removal of leading/trailing blanks.
 - CATS: concatenation with removal of leading/trailing blanks.
 - CATT: concatenation with removal of trailing blanks.
 - CATX: concatenation with removal of leading/trailing blanks with insertion of a separator.
- Parameters:
 - for CAT, CATS, CATT:
 - string1, ..., stringN
 - for CATX:
 - separator
 - string1, ..., stringN
- No more trim(left(var1))||'-'||trim(left(var2))... etc.

cat/cats/catt/catx warning

Beware – Trouble may be lurking. According to SAS help, the lengths of results vary depending on usage:

- 200 characters in WHERE clauses and in PROC SQL
- 32767 characters in the DATA step except in WHERE clauses
- 65534 characters when string is called from the macro processor



cat/cats/catt/catx – examples from SAS help

Function	Equivalent Code
CAT(OF X1-X4)	X1 X2 X3 X4
CATS(OF X1-X4)	TRIM(LEFT(X1)) TRIM(LEFT(X2)) TRIM(LEFT(X3)) TRIM(LEFT(X4))
CATT(OF X1-X4)	TRIM(X1) TRIM(X2) TRIM(X3) TRIM(X4)
CATX(SP, OF X1-X4)	TRIM(LEFT(X1)) SP TRIM(LEFT(X2)) SP TRIM(LEFT(X3)) SP TRIM(LEFT(X4))

choosec/choosen

- Returns a value from a list of choices
- Separate functions for character and numeric choice lists
- Parameters (index, value1, ..., valueN)

- Examples:

* if the following code is submitted *;

```
data _null_;
```

```
  item=choosen(-5,1,2,3,4,5);
```

```
  code=choosec(3,'A1','B1','C1','D1');
```

```
  put item= code=;
```

```
run;
```

* the following would appear in the log: *;

```
item=1 code=C1
```

count

- Used to count occurrences of a string within another string
- Parameters:
 - string (source)
 - substring (excerpt)
 - modifiers:
 - i = ignore case (not the default)
 - t = trim trailing blanks

count – examples from SAS help

SAS Statements	Results
<pre>xyz='This is a thistle? Yes, this is a thistle.'; howmanythis=count(xyz,'this'); put howmanythis;</pre>	3
<pre>xyz='This is a thistle? Yes, this is a thistle.'; howmanyis=count(xyz,'is'); put howmanyis;</pre>	6
<pre>howmanythis_i=count('This is a thistle? Yes, this is a thistle.' , 'this', 'i'); put howmanythis_i;</pre>	4
<pre>variable1='This is a thistle? Yes, this is a thistle.'; variable2='is '; variable3='i'; howmanyis_i=count(variable1,variable2,variable3); put howmanyis_i;</pre>	4
<pre>expression1='This is a thistle? ' 'Yes, this is a thistle.'; expression2=kscan('This is',2) ' '; expression3=compress('i ' ' t'); howmanyis_it=count(expression1,expression2,expression3); put howmanyis_it;</pre>	6

countc

- Used to count occurrences of a character within a string
- Parameters:
 - string (source)
 - characters
 - modifiers:
 - i = ignore case during the count. (not the default).
 - o = processes characters and modifiers only once, at the first call to this instance of COUNTC. Consequently, if you change the value of characters or modifiers in subsequent calls, the change is ignored by COUNTC
 - t = trim trailing blanks from string and characters parameters.
 - v = will only count characters that are not in the 'characters' parameter.

countc - examples from SAS help

SAS Statements	Results
<pre>xyz='Baboons Eat Bananas '; howmanya=countc(xyz,'a'); put howmanya;</pre>	5
<pre>xyz='Baboons Eat Bananas '; howmanyb=countc(xyz,'b'); put howmanyb;</pre>	1
<pre>howmanyb_i=countc('Baboons Eat Bananas ','b','i'); put howmanyb_i;</pre>	3
<pre>xyz='Baboons Eat Bananas '; howmanyab_i=countc(xyz,'ab','i'); put howmanyab_i;</pre>	8
<pre>variable1='Baboons Eat Bananas '; variable2='ab'; variable3='iv'; howmanyab_iv=countc(variable1,variable2,variable3); put howmanyab_iv;</pre>	16
<pre>expression1='Baboons 'Eat Bananas '; expression2=trim('ab '); expression3=compress('i 'v ' t'); howmanyab_ivt=countc(expression1,expression2,expression3); put howmanyab_ivt;</pre>	11

ifc

- ifc (condition, value #1, value #2, value #3)
- looks at a condition and returns user-specified, character values if the condition is true, false, or results in a missing value.
- Parameters:
 - condition (numeric expression)
 - value #1: character expression if condition is true.
 - value #2: character expression if condition is false.
 - value #3: character expression if condition results in a missing value.

ifn

- ifn (condition, value #1, value #2, value #3)
- looks at a condition and returns user-specified, numeric values if the condition is true, false, or results in a missing value.
- Parameters:
 - condition (numeric expression)
 - value #1: num expression if condition is true.
 - value #2: num expression if condition is false.
 - value #3: num expression if condition results in a missing value.

ifn comparison to if/then/else

This:

```
com=ifn(sales >= quota, sales*mqpct,  
        sales*nmqpct);
```

is the same as:

```
if sales >= quota then  
    com=sales*mqpct;  
else  
    com=sales*nmqpct;
```

ifc/ifn example

```
data sales;
  input salesman $ sales quota mqpct nmqpct;
  com=ifn(sales >= quota, sales*mqpct, sales*nmqpct);
  put salesman= sales= quota= com=;
  datalines;
  Rob 30000 27000 .09 .05
  Brian 10000 15000 .09 .025
  Ed 11000 15000 .05 .025
  Jim 15000 15000 .05 .025
  ;
run;

title 'Met Sales Quotas';
proc print data=sales;
  where ifc(sales >= quota, 'met', 'not met') eq 'met';
  var salesman sales quota com mqpct nmqpct;
run;
```

lengthc, lengthm, lengthn

- Differences:
 - lengthc: returns the length of a string including trailing blanks.
 - lengthm: returns the length of the variable allocated in memory.
 - lengthn: returns the length of a string excluding the trailing blanks.
 - returns 0 if a string is blank, compared to the *length* function which returns a 1.

propcase

- Takes a string, changing all characters to lower case, except for those following a delimiter.
- Default delimiters: space, slash, hyphen, period, tab, open parenthesis.

scanq

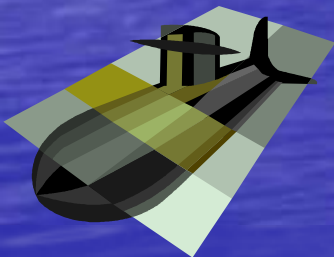
- Same thing as the *scan* function except delimiters within quotation marks are ignored. What's within quotes will not be picked apart.
- Both single and double quotation marks cannot be passed in the delimiter parameter of the function.

strip

- Combines the functions *trim* and *left*
- Can return a null value (0 length).

subpad

- parameters:
 - string – string to take an excerpt (substring) from.
 - position – location of first character in the substring.
 - length – the number of characters in the
- *subpad* WILL return a variable with the length specified, padding the results with spaces.
- *subpad* can return a string with a length of zero.



substrn

- parameters:
 - string – string to take an excerpt (substring) from.
 - position – location of first character in the substring.
 - length – the number of characters in the
- same as the *substr* function except:
 - *substrn* truncates the result when length exceeds the length of the string.
 - there will be no error messages for invalid third arguments.

substr vs. subpad vs. substrn

```
data test;  
  retain string "abcd";  
  drop string;  
  do Position = -1 to 6;  
    do Length = max(-1,-position) to 7-position;  
      str = substr(string, position, length);  
      pad = subpad(string, position, length);  
      strn = substrn(string, position, length);  
      lencstr = lengthc(str);  
      lencpad = lengthc(pad);  
      lencstrn = lengthc(strn);  
      lenmstr = lengthm(str);  
      lenmpad = lengthm(pad);  
      lenmstrn = lengthm(strn);  
      lennstr = lengthn(str);  
      lennpad = lengthn(pad);  
      lennstrn = lengthn(strn);  
      output;  
    end;  
  end;  
  
proc print; run;
```

Wrap up...

- ANYxxxx/NOTxxxx
- cat/cats/catt/catx
- choosec/chooseen
- count/countc
- ifc/ifn
- lengthc/lengthm/
lengthn
- propcase
- scanq
- strip
- subpad
- substrn

References

- SAS 9 – SAS Help and Documentation

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Questions...

