30th July 2013

String Functions In Datastage

String functions

Where Its Used: Transformer Stage(Derivation,Stage Variable,Loop Variable,Constraint in Transformer)

You might also like:

ETL Design essentials [http://www.datawarehousing-praveen.com/2015/03/etl-design-essentials.html]

For Datastage Date and Time Functions [http://www.datawarehousing-praveen.com/2012/10/date-and-time-functions-you-can-use.html]

Null Handling Functions In Datastage Transformer [http://www.datawarehousing-praveen.com/2013/07/null-handling-functions-in-datastage.html]

IBM Datastage Admin Commands Part 1 [http://www.datawarehousing-praveen.com/2013/06/ibm-datastage-admin-commands-part-1.html]

Datastage Interview Questions-Part 1 [http://www.datawarehousing-praveen.com/2013/10/datastage-interview-questions-part-1_720.html]

AlNum

Checks whether the given string contains only alphanumeric characters.

- **Input**: string (string)
- **Output**: true/false (int8)
- **Examples**. If mylink.mystring1 contains the string "OED_75_9*E", then the following function would return the value -1 (false).

  AlNum(mylink.mystring1)

  If mylink.mystring2 contains the string "12 red roses", then the following function would return the value 1 (true).

  AlNum(mylink.mystring2)

Alpha

Checks whether the given string contains only alphabetic characters.

- **Input**: string (string)
- **Output**: true/false (int8)
- **Examples**. If mylink.mystring1 contains the string "12 red roses", then the following function would return the value -1(false).

  Alpha(mylink.mystring1)

  If mylink.mystring2 contains the string "twelve red roses", then the following function would return the value 1 (true).

  Alpha(mylink.mystring2)

CompactWhiteSpace

Return the string after reducing all consecutive white space to a single space.

- **Input**: string (string)
- **Output**: result (string)
- **Examples**. If mylink.mystring contains the string "too many spaces", then the following function returns the string "too many spaces":

CompactWhiteSpace(mylink.mystring)

Compare

Compares two strings for sorting. The comparison can be left-justified (the default) or right-justified. A right-justified comparison compares numeric sub strings within the specified strings as numbers. The numeric strings must occur at the same character position in each string.

For example, a right-justified comparison of the strings AB100 and AB99 indicates that AB100 is greater than AB99 since 100 is greater than 99. A right-justified comparison of the strings AC99 and AB100 indicates that AC99 is greater since C is greater than B.

- **Input**: string1 (string), string2 (string), [justification (L or R)
- **Output**: result (int8), can be -1 for string1 is less than string2, 0 for both strings are the same, 1 for string1 is greater than string2.
- **Examples**. If mylink.mystring1 contains the string "AB99" and mylink.mystring2 contains the string "AB100", then the following function returns the result 1.

Compare(mylink.mystring1,mylink.mystring2,L)

If mylink.mystring1 contains the string "AB99" and mylink.mystring2 contains the string "AB100", then the following function returns the result -1.

Compare(mylink.mystring1,mylink.mystring2,R)

CompareNoCase

Compares two strings for sorting, ignoring their case.

- **Input**: string1 (string), string2 (string)
- **Output**: result (int8), can be -1 for string1 is less than string2, 0 for both strings are the same, 1 for string1 is greater than string2.
- **Examples**.

If mylink.mystring1 contains the string "Chocolate Cake" and mylink.mystring2 contains the string "chocolate cake", then the following function returns the result.

ComparNoCase(mylink.mystring1,mylink.mystring2)

CompareNum

Compares the first *n* characters of two strings.

- **Input**: string1 (string), string2 (string), length (int16)
- **Output**: result (int8), can be -1 for string1 is less than string2, 0 for both strings are the

same, 1 for string1 is greater than string2.

- **Examples**. If mylink.mystring1 contains the string "Chocolate" and mylink.mystring2 contains the string "Choccy Treat", then the following function returns the result 1.

```
ComparNum(mylink.mystring1,mylink.mystring2,4)
```

CompareNumNoCase

Compares the first *n* characters of two strings, ignoring their case.

- **Input**: string1 (string), string2 (string), length (int16)
- **Output**: result (int8), can be -1 for string1 is less than string2, 0 for both strings are the same, 1 for string1 is greater than string2.
- **Examples**.
  If mylink.mystring1 contains the string "chocolate" and mylink.mystring2 contains the string "Choccy Treat", then the following function returns the result 1.

```
ComparNumNoCase(mylink.mystring1,mylink.mystring2,4)
```

Convert

Converts characters in the string designated in *expression*. Converts the characters specified in *fromlist* to the characters specified in *tolist*.

- **Input**: fromlist (string), tolist (string), expression (string)
- **Output**: result (string)
- **Examples**. If mylink.mystring1 contains the string "NOW IS THE TIME", then the following function returns the string "NOW YS XHE XYME".

```
Convert("TI","XY",mylink.mystring1)
```

Count

Counts the number of times a substring occurs in a string.

- **Input**: string (string), substring (string)
- **Output**: result (int32)
- **Examples**. If mylink.mystring1 contains the string "chocolate drops, chocolate ice cream, chocolate bars", then the following function returns 3.

```
Count(mylink.mystring1,"choc")
```

Count

Counts the number of delimited fields in a string.

- **Input**: string (string), delimiter (string)
- **Output**: result (int32)
- **Examples**.
  If mylink.mystring1 contains the string "chocolate drops, chocolate ice cream, chocolate bars", then the following function returns 3.

```
Dcount(mylink.mystring1,",")
```

DownCase

Changes all uppercase letters in a string to lowercase.

- **Input**: string (string)
- **Output**: result (string)
- **Examples**. If mylink.mystring1 contains the string "CaMel cAsE", then the following function returns the string "camel case".

```
DownCase(mylink.mystring1)
```

DQuote

Encloses a string in double quotation marks.

- **Input**: string (string)
- **Output**: result (string)
- **Examples**. If mylink.mystring1 contains the string needs quotes, then the following function returns the string "needs quotes".

```
DQuote(mylink.mystring1)
```

Field

Returns one or more substrings located between specified delimiters in a string. The argument *occurrence* specifies which occurrence of the delimiter is to be used as a terminator.

The argument *number* optionally specifies how many substrings to return.

- **Input**: string (string), delimiter (string), occurrence (int32), [number (int32)]
- **Output**: result (string)
- **Examples**. If mylink.mystring1 contains the string "chocolate drops, chocolate ice cream, chocolate bars, chocolate dippers", then the following function returns the string "chocolate ice cream".

```
Field(mylink.mystring1,",",2)
```

If mylink.mystring1 contains the string "chocolate drops, chocolate ice cream, chocolate bars, chocolate dippers", then the following function returns the string "chocolate ice cream, chocolate bars".

```
Field(mylink.mystring1,",",2,2)
```

Index

Finds the starting character position of a substring. The argument *occurrence* specifies which occurrence of the substring is to be located.

- **Input**: string (string) substring (string) occurrence (int32)
- **Output**: result (int32)
- **Examples**. If mylink.mystring1 contains the string "chocolate drops, chocolate ice cream,

chocolate bars, chocolate dippers", then the following function returns the value 18.

```
Index(mylink.mystring1,"chocolate",2)
```

Left

Returns the leftmost *n* characters of a string.

- **Input**: string (string) number (int32)
- **Output**: result (string)
- **Examples**.

  If mylink.mystring1 contains the string "chocolate drops, chocolate ice cream, chocolate bars, chocolate dippers", then the following function returns the string "chocolate".

```
Left(mylink.mystring1,9)
```

Len

Returns the length of a string in characters.

- **Input**: string (string)
- **Output**: result (int32)
- **Examples**. If mylink.mystring1 contains the string "chocolate", then the following function returns the value 9.

```
Len(mylink.mystring1)
```

Num

Returns 1 if string can be converted to a number, or 0 otherwise.

- **Input**: string (string)
- **Output**: result (int32)
- **Examples**. If mylink.mystring1 contains the string "22", then the following function returns the value 1.

```
Num(mylink.mystring1)
```

If mylink.mystring1 contains the string "twenty two", then the following function returns the value 0.

```
Num(mylink.mystring1)
```

PadString

Return the string padded with the specified number of pad characters.

- **Input**: string (string) padstring (string) padlength (int32)
- **Output**: result (string)
- **Examples**. IIf mylink.mystring1 contains the string "AB175", then the following function returns the string "AB17500000".

PadString(mylink.mystring1,"0",5)

Right

Returns the rightmost *n* characters of a string.

- **Input**: string (string) number (int32)
- **Output**: result (string)
- **Examples**. If mylink.mystring1 contains the string "chocolate drops, chocolate ice cream, chocolate bars, chocolate dippers", then the following function returns the string "dippers".

Right(mylink.mystring1,7)

Soundex

Returns a code which identifies a set of words that are (roughly) phonetically alike based on the standard, open algorithm for SOUNDEX evaluation.

- **Input**: string (string)
- **Output**: result (string)
- **Examples**. If mylink.mystring1 contains the string "Griffin" then the following function returns the code "G615".

Soundex(mylink.mystring1)

If mylink.mystring1 contains the string "Griphin" then the following function also returns the code "G615".

Soundex(mylink.mystring1)

Space

Returns a string of *n* space characters.

- **Input**: length (int32)
- **Output**: result (string)
- **Examples**. If mylink.mylength contains the number 100, then the following function returns a string that contains 100 space characters.

Space(mylink.mylength)

SQuote

Encloses a string in single quotation marks.

- **Input**: string (string)
- **Output**: result (string)
- **Examples**. If mylink.mystring1 contains the string needs quotes, then the following function returns the string 'needs quotes'.

SQuote(mylink.mystring1)

Str

Repeats a string the specified number of time.

- **Input**: string (string)
- **Output**: result (string)
- **Examples**. If mylink.mystring1 contains the string needs "choc", then the following function returns the string "chocchocchocchocchoc".

Str(mylink.mystring1,5)

StripWhiteSpace

Returns the string after removing all whitespace characters from it.

- **Input**: string (string) repeats (int32)
- **Output**: result (string)
- **Examples**. If mylink.mystring contains the string "too many spaces", then the following function returns the string "toomanyspaces":

StripWhiteSpace(mylink.mystring)

Trim

Remove all leading and trailing spaces and tabs plus reduce internal occurrences to one. The argument*stripchar* optionally specifies a character other than a space or a tab. The argument *options* optionally specifies the type of trim operation to be performed and contains one or more of the following values:

A Remove all occurrences of *stripchar*

B Remove both leading and trailing occurrences of *stripchar*

D Remove leading, trailing, and redundant white-space characters

E Remove trailing white-space characters

F Remove leading white-space characters

L Remove all leading occurrences of *stripchar*

R Remove leading, trailing, and redundant occurrences of *stripchar*

T Remove all trailing occurrences of *stripchar*

- **Input**: string (string) [stripchar (string)] [options (string)]
- **Output**: result (string)
- **Examples**. If mylink.mystring contains the string " String with whitespace ", then the following function returns the string "String with whitespace":

Trim(mylink.mystring)

If mylink.mystring contains the string "..Remove..redundant..dots....", then the following function returns the string "Remove.redundant.dots":

```
Trim(mylink.mystring,".")
```

If mylink.mystring contains the string "Remove..all..dots....", then the following function returns the string "Removealldots":

```
Trim(mylink.mystring,".","A")
```

If mylink.mystring contains the string "Remove..trailing..dots....", then the following function returns the string "Remove..trailing..dots":

```
Trim(mylink.mystring,".","T")
```

TrimB

Removes all trailing spaces and tabs from a string.

- **Input**: string (string)
- **Output**: result (string)
- **Examples**. If mylink.mystring contains the string "too many trailing spaces ", then the following function returns the string "too many trailing spaces":

```
TrimB(mylink.mystring)
```

TrimF

Removes all leading spaces and tabs from a string.

- **Input**: string (string)
- **Output**: result (string)
- **Examples**. If mylink.mystring contains the string " too many leading spaces", then the following function returns the string "too many leading spaces":

```
TrimF(mylink.mystring)
```

UpCase

Changes all lowercase letters in a string to uppercase.

- **Input**: string (string)
- **Output**: result (string)
- **Examples**. If mylink.mystring1 contains the string "CaMel cAsE", then the following function returns the string "CAMEL CASE".

```
UpCase(mylink.mystring1)
```

TrimLeadingTrailing

Removes all leading and trailing spaces and tabs from a string.

- **Input**: string (string)
- **Output**: result (string)

- **Examples.** If mylink.mystring contains the string " too many spaces ", then the following function returns the string "too many spaces":

  TrimLeadingTrailing(mylink.mystring)

You might also like:

ETL Design essentials [http://www.datawarehousing-praveen.com/2015/03/etl-design-essentials.html]
For Datastage Date and Time Functions [http://www.datawarehousing-praveen.com/2012/10/date-and-time-functions-you-can-use.html]
Null Handling Functions In Datastage Transformer [http://www.datawarehousing-praveen.com/2013/07/null-handling-functions-in-datastage.html]
IBM Datastage Admin Commands Part 1 [http://www.datawarehousing-praveen.com/2013/06/ibm-datastage-admin-commands-part-1.html]
Datastage Interview Questions-Part 1 [http://www.datawarehousing-praveen.com/2013/10/datastage-interview-questions-part-1_720.html]

Posted 30th July 2013 by praveen govind

Labels: Datastage Functions

0 *Add a comment*