IDEA Data Analysis Software

Common @Functions

FUNCTIONS	FUNCTIONALITY	EXAMPLES
@age( )	Returns the number of days between two specified dates. Note: both dates must be a "Date" field type.	FieldName1 = "20111230" FieldName2 = "20111220" thus using: @age(FieldName1, FieldName2) yields the result: 10 (10 day difference) (Make sure that the latest date is FieldName1)
@compif()	A conditional statement which tests for multiple conditions and provides results depending on the outcome. Very helpful when stringing together multiple "if" statements.	FieldName= "NAME" thus using: @compif(NAME=="Bob", 1, NAME=="Tom", 2, NAME=="Ann", 3) will return 1, 2 or 3 depending on if the NAME field contains Bob, Tom or Ann.
@ctod( )	Converts a field containing dates stored as a character field type into a Date field type in IDEA's Date format.	FieldName = "12/20/11" thus using: @ctod(FieldName, "MM/DD/YY") yields the result: "20111220" (difference is this is now a date field type and no longer a character string and is accepted in functions requiring Date arguments.) Note the order of YYYYMMDD with the absence of extraneous separators. IDEA stores dates in this fashion, though they will appear as MM/DD/YYYY.
@getnextvalue( )	Returns the value of the next record from the specified field, starting with the first record.	ColumnOne = 1, 2, 3, 4, 5 | ColumnTwo = @getnextvalue("ColumnOne"), so it will contain values: 2, 3, 4, 5, 0 (For parallel columns, this will fill each row of the current column with the value of the row below from the corresponding column.)
@getpreviousvalue( )	Returns the value in the preceding record for the specified field, starting with the first record.	ColumnOne = 1, 2, 3, 4, 5 | ColumnTwo = @getpreviousvalue("ColumnOne"), so it will contain values: 0, 1, 2, 3, 4 (For parallel columns, this will fill each row of the current column with the value of the row above it from the corresponding column.)
@if( )	Tests a specified condition and if the result is true then it will return the first result (true), otherwise it returns the second result (false).	@if(10 > 6, "10 is larger than 6", "10 is less than 6") yields the result: "10 is larger than 6." First part accepts the values to test with a relational operator in between (>, <, =, ==, <>, <=, >=), second part is the result if the test is true (in this case 10 is greater than 6, so it outputs the second item.), last part results if the test were to return false.
@isini( )	Searches for the occurrence of a specified character string (of any case) in a character field or date field. If appending a field and the string is found, it returns the starting position of the specified character string; otherwise, a 0 is returned. If used in an extraction, the record is extracted if the character string is found.	FieldName = "sample string" thus using: @isini("mpl ", FieldName) in an appended field yields the result: 3 (determined the position where the string began); in an extraction the record is extracted.
@justletters( )	Returns characters, spaces and special characters from the character string or field by removing any numbers.	FieldName = "23W45 E% $R#T" thus using: @justletters(FieldName) will yield: "W E% $R#T"
@justnumbers( )	Returns only the numbers from the specified character string or field.	FieldName = "W987DTR^3&*%" thus using: @justnumbers(FieldName) will yield: 9873

If you get stuck along the way, use our "20 Minute Rule," which is designed to save you time and effort. If it takes you more than 20 minutes to utilize any IDEA function or feature, contact us for assistance - 888.641.2800 Option 4 • helpdesk@audimation.com

Learn more about IDEA and IDEA training opportunities at audimation.com.

FUNCTIONS	FUNCTIONALITY	EXAMPLES
@left( )	Returns the specified number of leftmost characters from the beginning of the character field.	FieldName = "word" thus using: @left(FieldName, 2) will yield: "wo"
@len( )	Reviews a character expression and returns the length of the character string, including spaces.	FieldName = "this sentence has 31 characters" thus using: @len(FieldName) will yield: 31
@lower( )	Converts a character field into lower case.	FieldName = "UPPER CASE" thus using: @lower(FieldName) will yield: "upper case"
@mid( )	Extracts a sub-string of a character field. You will need to specify the character field, the starting position of the sub-string within the character field and the number of characters.	FieldName = "my phrase" thus using: @mid(FieldName, 2, 6) will yield: "y phra"
@reverse( )	Reverses a string of characters.	FieldName = "Left to right" thus using: @reverse(FieldName) will yield: "thgir ot tfeL"
@right( )	Identifies the specified number of rightmost characters in the character field.	FieldName = "word" thus using: @right(FieldName, 2) will yield: "rd"
@spacestoone( )	Removes all extra spaces from a cell, leaving a single space between phrases.	FieldName = " too many spaces " thus using: @spacestoone(FieldName) will yield: "too many spaces"
@spanexcluding( )	Returns the characters in a field from the first character up to, but not including, the character specified.	FieldName = "My Sentence" thus using: @spanexcluding(FieldName, " ") will yield: "My"
@simplesplit( )	Breaks a character string into segments by breaking on a specified character or character string(such as :,/,-,...) within the string and returns a specified segment.	FieldName= "123-45-6789" thus using: @simplesplit(FieldName,"-",1,"-") will yield "45". The first "-" tells it what the starting separator is. The 1 tells it to grab the first segment that starts with a "-" and ends with a "-". The second "-" tells it what the ending separator is.
@str( )	Converts a number stored as a numeric data type into a character data type.	FieldName = 12345 thus using: @str(FieldName,0,0) will yield: "12345". The first 0 is the minimum length of the string and the second 0 is the decimals required. The content will be left justified in the new field.
@strip( )	Removes all spaces, punctuation and special characters from a text field leaving a string of characters and numbers.	FieldName = "-109A B,V=$%" thus using: @strip(FieldName) will yield: "109ABV"
@upper( )	Converts a character field into upper case.	FieldName = "lower case" thus using: @upper(FieldName) will yield: "LOWER CASE"
@val()	Converts a number stored as a character data type to a numeric data type. Useful when a field contains numbers which need to be used to perform calculations.	FieldName="123AB4" (as a character field) thus using: @val(FieldName) will yield: "123" in a numeric field.