Formulas in Datorama

Modified on: Mon, 4 Oct, 2021 at 10:00 AM

Contents

Number Formulas
Text Formulas
Date Formula
Aggregation Formulas
Logical Formulas
Type Conversion

This article describes the available formulas that can be used when you are creating Calculated Dimensions and/or Measurements or when you are mapping a Data Stream.

Number Formulas

ABS
Syntax: (number)
Description: Returns the absolute value of a number
Example:
ABS(1) = 1
ABS(-1) = 1

CEILING
Syntax: (number, significance)
Description: Returns number rounded up, away from zero, to the nearest significance.
Example:
CEILING(2.1,1) = 3
CEILING(2.8,1) = 3
If the significance is higher than 1, the output will be equal to the closest multiple of the significance that is higher than the number:
CEILING(17.1,5) = 20
CEILING(51.1,20) = 60

FLOOR
Syntax: (number, significance)
Description: Rounds the number down, towards zero, to the nearest significance.
Example:
FLOOR(2.1,1) = 2
FLOOR(2.8,1) = 2
If the significance is higher than 1, the output will be equal to the closest multiple of the significance that is smaller than the number:
FLOOR(17.1,5) = 15
FLOOR(51.1,20) = 40

RAND
Syntax: ()
Description: Gets a random number between 0 and 1.
Does not require an input. Always returns 16 digits after the decimal.
Example:
RAND() = 0.5154096042646703
RAND() = 0.9764887924341263

ROUND
Syntax: (number)
Description: Rounds a number up or dow to the nearest integer.
Example:
ROUND(3.4) = 3
ROUND(3.5) = 4

Text Formulas

CHAR
Syntax: (number)
Description: Returns the character specified by a number.
Example:
CHAR(65) = A
CHAR(67) = C
CHAR(63) = ?
For more information on the ASCII table, **click here (http://www.asciitable.com/)**.

COALESCE
Syntax: ([value1, value2, ...])
Description: Returns the first value from the list that is not empty.
Example:
Mainly used in the mapping of a Data Stream when more than one field is defined for a single item. For example, in the original Data Stream "Campaign Key" was mapped, but now the field is called "Campaign ID". To search for both "Campaign Key" and "Campaign ID", go to the Data Stream mapping screen and in the formula Box for the "Campaign Key" field, enter the following syntax:
COALESCE([csv[`Campaign Key`], csv[`Campaign ID`]])
The syntax means, first call the field "Campaign Key", if no value is found, "Campaign ID" should be used in its place.
For more information about this formula, **click here. (https://support.datorama.com/en/support/solutions/articles/4000084033-coalesce-function)**

CODE
Syntax: (text)
Description: Returns a numeric code for the first character in a text string.
Example:
The opposite of the CHAR formula.
CODE('A') = 65
CODE('C') = 67

CODE('?') = 63
For more information on the ASCII table, **click here (http://www.asciitable.com/)**.

COSTTYPE
Note: This is only available in the Data Stream Mapping
Syntax: (text)
Description: This formula matches between the Custom Cost Type in the source data, and the Custom Cost Types in Datorama that are set in the Cost Center section under Workspace settings.
Example:
COSTTYPE(csv[Cost Type'])
COUNTRY_TO_ISO2

COUNT_CHAR
Note: This formula is case-sensitive
Syntax: (text,charToCount)
Description: Counts how many times a specific character appears in a text.
Example:
COUNT_CHAR('Datorama','a') = 3
COUNT_CHAR('DatoramA','a') = 2

EXTRACT
Syntax: (text, delimiter, position)
Description: Extracts a substring from a given text at a given position, after splitting the string by the given delimiter.
Example:
EXTRACT(csv['Media Buy Name'], '_',1);
EXTRACT('a_b_c', '_', 1) will return 'b'
Explanation: In this example, the formula will look for the Media Buy Name, and will then EXTRACT the substring from position 1 (which is the 2nd position, since the index begins at the number 0). The positions are defined by the delimiter, which in this example is '_'.

FIND
Syntax: (findText, withinText)
Description: Returns the starting position of findText in withinText. Returns -1 if withinText does not contain findText, The count starts at 0.
Example:
FIND('D', 'Datorama') = 0
FIND('a', 'Datorama') = 1
FIND('Z', 'Datorama') = -1

FUZZYMATCH
Syntax: (valueToMatch,array,threshold)
Description: Searches for a string in an array of values that potentially match using duplicate-detection algorithms that calculate the similarity of two streams of data. Returns the matched string from the array in case of a match, or an empty string if no match.
Example:

FUZZYMATCH("ABC Compani",{"ABC Company Inc","Apple Inc","IBM Company"},0.8) will return "ABC Company Inc". Note that the threshold parameter must be a value between 0 and 1. Threshold of 1 means an exact match.
FUZZYMATCH('Camp',{[Media_Buy_Name]},0.8)
The array can be a mapped Dimension such as the [Media_Buy_Name], but it has to be wrapped in curly brackets in order to indicate that this is an array. The value to match can be another mapped Dimension.

FUZZYVLOOKUP
Note: This is only available in the Data Stream Mapping
Syntax: (valueToSearch, searchDimension, returnDimension, threshold, workspaceLevel, {'streamfilter1', 'streamfilter2', ...}, useAutomaton)
Description: The FUZZYVLOOKUP formula uses the value provided as the first parameter and searches for it in the Dimension provided as the second parameter using a **FUZZYMATCH**. Once found, it returns the value of the Dimension used as the third parameter.

Note that both the lookup Dimension and the output Dimension must be related to the same entity. workspaceLevel - Set the value to 'true' if you want to search through all the Workspace's data, or set 'false' to search in the current Data Stream only.
The threshold parameter must be a value between 0 and 1 wherea threshold of 1 means an exact match.
streamfilter (optional) – Enterthe Data Stream Names you want to apply the formula to inside the {curly brackets}, instead of 'streamfilter#'. If 'searchDimension' contains more than 2,000 unique values, the formula will fail.
useAutomaton - Forces the use of the Levenshtein automaton algorithm, which supports very large dictionary sizes.
Example:
FUZZYVLOOKUP(csv['Key'] ,[Main_Lookup_Levels_Entity_Key], [Main_Lookup_Levels_Entity_Name], 0.9, true, {'Lookup Table Level 1'}, true)

GET_EXCHANGE_RATE
Note: This is only available in the Data Stream Mapping
Syntax: (date, fromISO, toISO)
Description: Gets the exchange rate for 2 currencies within a specific date.
Example: GET_EXCHANGE_RATE(DATE('2015','8',17),'USD','GBP').
This formula returns the rate for the certain date selected.
If you want to retrieve more specific date information, the following formulas can be used along with Get Exchange Rate: To choose a certain hard-coded date use the DATE formula as follows:
GET_EXCHANGE_RATE(DATE(2015,5,17),'USD','GBP') * csv['media_cost'];
To dynamically change dates use either the TODAY, YESTERDAY, etc., formulas as follows:
GET_EXCHANGE_RATE(TODAY(), 'USD', 'GBP') * csv['revenue'];
To get the source file dates, by input the relevant csv field as follows:
GET_EXCHANGE_RATE(csv['date'], 'USD', 'GBP') * csv['media_cost'].
To remove about this formula, **click here (https://support.datorama.com/en/support/solutions/articles/4000084032-get-exchange-rate-function)**.

GET_VAR

Note: This is only available in the Data Stream Mapping

Syntax: (name)

Description: The GET_VAR formula can be used for data files that contain a certain value that appears once and it applies the value to all rows beneath it.

To learn more about this formula, **click here (https://support.datorama.com/en/support/solutions/articles/4000084031-get-var-set-var-functions)**.

Example:

```
var camp = csv['Campaign'];

if(!ISEMPTY(camp)) {
SET_VAR('pick_a_name',camp);
return camp;
} else {
return GET_VAR('pick_a_name');
}
```

GROUPCONCAT

Syntax: (groupedVal, groupKey, separator)

Description: Concatenates the groupedVal by the given groupKey, with the separator string in-between the grouped values. This is done uniquely if the same value will repeat itself more than once per the groupKey - it will not be concatenated twice. The concatenated values will be sorted in alphabetical order.

This is extremely useful when there is a need to query a value, and there is more than one match. This formula will allow to include all the matches found.

Example:

GROUPCONCAT(csv['Ad Name'],csv['Campaign Name'],'-')

Explanation: This will result in a Dimension value that contains all the Ad Names that are linked to a certain Campaign Name concatenated without repetition.

To learn more about this formula, **click here (https://support.datorama.com/en/support/solutions/articles/4000084035-groupconcat)**.

INDEXOF

Syntax: (array,findText)

Description: Searches the array for a specified item, and return its position. Returns -1 if the item is not found.

If the item appears more than once, the INDEXOF formula returns the position of the first occurrence.

Example:

INDEXOF(csv['row'], 'Campaign ID') == 0 will return true if 'Campaign ID' was found in the first cell.

Note: The first item has a position of 0, the second item has position 1, and so on.

INDEXOF({csv['Placement Name']},'ABC') == 0

Returns true for all rows of data from the source file, where the csv['Placement Name'] value is equal to 'ABC'.

ISEMPTY
Syntax: (value)
Description: Returns whether the value is empty or not.
Example:
ISEMPTY(csv['Placement Name'])
ISEMPTY([Media_Buy_name])
Note: Adding an exclamation mark at the beginning of your query reverses the meaning of the formula and returns true if the field is not empty as follows:
!ISEMPTY(csv['Placement Name'])

LEFT
Syntax: (text, numOfChars)
Description: Returns the requested number of characters starting from the left.
Example:
LEFT('Datorama', 4) = 'Dato'

LEN
Syntax: (text)
Description: Returns the number of characters in the given text.
Example:
LEN('Datorama') = 8

LOWER
Syntax: (text)
Description: Converts all uppercase letters in a text string to lowercase.
Example:
LOWER('DATORAMA') = 'datorama'

MATCH
Syntax: (valueToMatch, array)
Description: Searches for a specified item in an array of values, and then returns a boolean result.
Example:
MATCH('ABC',['ABC', 'LMN', 'XYZ']) = true
MATCH('DDD',['ABC', 'LMN', 'XYZ']) = false

PROPER
Syntax: (text)
Description: Capitalizes the first letter in a text string and any other letters in the text that follow a character other than a letter and converts all other letters to lowercase letters.
Example:
PROPER('frank lampard') = Frank Lampard

REPLACE
Syntax: (sourceStr, pattern, targetStr)
Description: Replaces every match of a pattern (it could also be a regular expression) in the source string (sourceStr) to a target string (targetStr).
Example:

REPLACE(csv['Campaign Name'], '-', ' ')
Explanation: In the Campaign Name, the formula will take every instance of "-" and replace with a space bar.
Example: REPLACE(csv['media buy key'],'\\s','-')
Explanation: All white-space characters (for example, the space bar) will be replaced with a hyphen (-), from 'media buy 1' to 'media-buy-1'. All regular expressions that contain backslashes (e.g. \s, \d, \w) should be escaped (add a backward slash before the symbol) which is why we enter \\s in the REPLACE formula, and not just \s.
Learn more on the REPLACE formula **here.** **(https://support.datorama.com/en/support/solutions/articles/4000084034-the-replace-function)**

RIGHT
Syntax: (text, numOfChars)
Description: Returns the requested number of characters from the right side of the string.
Example:
RIGHT('ABCD',2) = 'CD'

SELECT
Syntax: (sourceStr, pattern)
Description: Selects the first match of the pattern (a regular expression) in the sourceStr.
Example:
SELECT('Hello B wow E Goodbye', 'B.*?E') would return 'B wow E'.
Advanced example:
SELECT('ABC 25-Dec-2018 XYZ','\\d{2}-\\w{3}-\\d{4}') = '25-Dec-2018'
Explanation: In this case, Datorama will look for 2 digits (\\d{2}) and then 3 letters (\\w{3}) and 4 more digits (\\d{4})
SELECT([Media_Buy_Name],'\\d{2}-\\w{3}-\\d{4}')

SET_VAR
Syntax: (name,value)
Description: Sets a global variable value.
Example:
Refer to **GET_VAR** formula.
Learn more about this formula **here** **(https://support.datorama.com/en/support/solutions/articles/4000084031-get-var-set-var-functions)**.

SUBSTITUTE
Syntax: (text, oldText, newText)
Description: Substitutes oldText with newText in a text string. Use SUBSTITUTE when you want to replace specific text in a text string.
Note: This formula is not compatible with regular expressions.
Example:
SUBSTITUTE('ABC', 'A', 'Z') = 'ZBC'
SUBSTITUTE([Media_Buy_Name], 'A', 'Z')

SUBSTRING
Syntax: (text, beginIndex, endIndex)

Description: Returns the substring of the text beginning at beginIndex (inclusive), and ending at endIndex (excluding). The first character is represented by index 0.
Example:
SUBSTRING('ABCD',0,1) = 'A'
SUBSTRING('ABCD',0,2) = 'AB'
SUBSTRING('ABCD',1,2) = 'B'
SUBSTRING([Campaign_Name],1,2)

TARGETTYPE
Syntax: (text)
Description: This formula matches between the custom target type in the source data to the custom target type in Datorama.
Example:
TargetType(csv['Custom Target Type'])

TRIM
Syntax: (text)
Description: Removes all spaces from the text except for single spaces between words.
Example:
TRIM(' A B C ') = 'A B C'
TRIM([Creative_Name])

UPPER
Syntax: (text)
Description: Converts all lowercase letters in a text string to uppercase letters.
Example:
UPPER('abc') = 'ABC'
UPPER([Campaign_Name])

URL_DECODE
Syntax: (text)
Description: Decodes an application/x-www-form-urlencoded string using a UTF-8 encoding scheme.
Example:
URL_DECODE('this+is+some+random+text') = 'this is some random text'
Click here (https://www.url-encode-decode.com/) to learn more

URL_ENCODE
Syntax: (text)
Description: Translates text into application/x-www-form-urlencoded format using a UTF-8 encoding scheme.
Example:
URL_ENCODE('this is some random text') = 'this+is+some+random+text'
Click here (https://www.url-encode-decode.com/) to learn more

VLOOKUP
Syntax: VLOOKUP(valueToSearch, searchDimension, returnDimension, workspaceLevel, concatValues, {'streamFilter1', 'streamFilter2',...})

Description: The VLOOKUP formula uses the value provided as the first parameter and searches it in the Dimension provided as the second parameter. Once found, it returns the value of the Dimension used as the third parameter. Note that both the lookup Dimension and the output Dimension must be related to the same entity. The fourth parameter should be set to True if you want to search in all the brand's data, or False if you want to search in the current Data Stream only.
Example:
VLOOKUP(csv['Placement Name'],[Media_Buy_Key],[Media_Buy_Name],true) will result in returning the Media Buy Key for each Media Buy Name found in the same Data Stream. The fifth and sixth parameters are optional - concatValues will indicate to concatenate multiple results found for this valueSearchDataStreams should contain a list of data streams between curly brackets {'streamFilter1', 'streamFilter2', ...}, and will indicate where the values will be searched.
Learn more about this formula **here (https://support.datorama.com/en/support/solutions/articles/4000087823-vlookup)**.

VLOOKUP UNNEST
Syntax: VLOOKUP_UNNEST(valueToSearch, searchDimension, returnDimension, workspaceLevel, {'streamFilter1', 'streamFilter2',...})
Description: VLOOKUP_UNNEST is a formula that works the same as a regular VLOOKUP, except for when having multiple results in the **returnDimension**, it will return them in separate rows, instead of either the first value found, or concatenated (To be determined in the concatValues field). VLOOKUP_UNNEST will break the return Dimensions into separate rows automatically.
Note: Vlookup Unnest adds more rows to your Workspace than a regular Vlookup, as it separates multiple return values into different rows.
Learn more about this formula **here (https://support.datorama.com/en/support/solutions/articles/4000087823-vlookup#vlookupunnest)**.

Date Formulas

DATE
Syntax: (year, month)
Description: Returns the 1st date of the given month in the given year.
Note: Date formulas do not allow the use of a '0' prefix in the numbers 08 & 09. Be sure to insert numbers without a '0' prefix, e.g. '8' and not '08'.
Example:
DATE(2018,12) = Sat Dec 01 00:00:00 UTC 2018

DATE
Syntax: (year, month, day)
Description: Returns the date representing the given arguments.
Example:
DATE(2018,12,26) = Wed Dec 26 00:00:00 UTC 2018

DATEADD
Syntax: (interval, number, date)

Description: Returns the date after which a certain time/date interval has been added. The interval should be one of the following: 'yyyy' (year), 'q' (quarter), 'm' (month) , 'd' (day), 'ww' (week), 'h' (hour), 'n' (minute), 's' (second).
Note: Date formulas do not allow the use of a '0' prefix in the numbers 08 & 09. Be sure to insert numbers without a '0' prefix, e.g. '8' and not '08'.
Example:
DATEADD('m',-1,csv['date'])
Explanations: The output will be the given date but shifted one month back. So if the input is '2018-9-01' the output will be '2018-08-01'.

DATEDIFF
Syntax: (interval, startDate, endDate)
Description: Returns the number of full time units specified by 'interval' between the startDate and the endDate. The startDate and endDate arguments should be date objects or date strings in standard form.
The following intervals can be used: 'yyyy' (year), 'q' (quarter), 'm' (month) , 'd' (day), 'ww' (week), 'h' (hour), 'n' (minute), 's' (second).
If the difference between the two dates is less than 24 hours, the output will be 0 days.
If the difference between the two dates is less than 7 days, the output will be 0 weeks.
Examples:
DATEDIFF('d','2018-12-01 18:00:00','2018-12-02 09:00:00') = 0
(there are less than 24 hours between the 2 timestamps, so the formula correctly outputs that there are 0 full days between them)
DATEDIFF('d','2018-12-01','2018-12-02') = 1
(no timestamps, so the formula outputs 1 as it assumes the timestamp of both dates is 00:00:00)
DATEDIFF('ww','2018-12-01','2018-12-07') = 0
(the difference in days between these two dates is 6, which is why the formula outputs 0 as there is not a full week of a difference between them)
DATEDIFF('m','2018-12-01','2019-02-07') = 2
(2 full months and 6 days have passed between the start date and the end date, so the output is 2)
DATEDIFF('d',[Media_Buy_Start_Date],[Media_Buy_End_Date])

DAYPRECISION
Syntax: (date)
Description: Returns the given date with day precision, meaning hours/minutes/seconds fields are all set to 0.
For example the date '12-Jun-2018 16:34' will be converted to '12-Jun-2018 00:00'
Note: Date formulas do not allow the use of a '0' prefix in the numbers 08 & 09. Be sure to insert numbers without a '0' prefix, e.g. '8' and not '08'.
Example:
DAYPRECISION('2018-12-25 23:23:23') = '2018-12-25 00:00:00'

DAYSBETWEEN
Syntax: (startDate,endDate)
Description: Returns the number of days between the given startDate and endDate. The arguments should be date objects or date strings in standard form.

If the difference between the two dates is less than 24 hours, the output will be 0.
Example:
DAYSBETWEEN('2018-12-01','2018-12-02') = 1
DAYSBETWEEN([Media_Buy_Start_Date],[Media_Buy_End_Date])

DAYS_IN_MONTH
Syntax: (date)
Description: Returns the number of days in a specified month.
Example:
DAYS_IN_MONTH(TODAY()) = the number of days in the current month
DAYS_IN_MONTH([Day])

LAST_DATA_DATE
Syntax: ()
Description: Returns the last data date of the current Data Stream.

NETWORKDAYS
Syntax: (startDate,endDate)
Description: Returns the number of workdays between two dates, excluding weekends. The arguments should be date objects or date strings in a standard form.
Example:
NETWORKDAYS('2018-12-01','2018-12-31') = 20
NETWORKDAYS(DATE(2018,12),DATE(2018,12,31)) = 20

NOW
Syntax: ()
Description: Returns the current system date and time.
Example:
NOW() = 'Sun Dec 16 08:37:56 UTC 2018'

SETHOUR
Syntax: (date, hour)
Description: Returns the given date, set with the given hours, the hour being a numeric value between 0 and 23.
In order to use this formula, you have to first enable the 'Enable Hourly Breakdown' in the settings of the Data Stream. Then, in the Day field in the Data Stream mapping, insert the proper csv field and the proper hour input.
Note: Date formulas do not allow the use of a '0' prefix in the numbers 08 & 09. Be sure to insert numbers without a '0' prefix, e.g. '8' and not '08'.
Example:
SETHOUR(csv['date'],csv['hour'])

TODAY
Syntax: TODAY()
Description: Returns the current system date with a timestamp of 00:00:00.
Example:
TODAY() = 'Sun Dec 16 00:00:00 UTC 2018'

TOSECONDS
Syntax: (duration)

Description: Returns the number of seconds represented by the given duration string, with the following format: HH:mm:ss/HH:mm:ss.SSS.
Example:
TOSECONDS('00:02:30') = 150
TOSECONDS(csv['time'])

YESTERDAY
Syntax: YESTERDAY()
Description: Returns yesterday's system date with a timestamp of 00:00:00.
Example:
TODAY() = 'Sun Dec 15 00:00:00 UTC 2018'

Aggregation Formulas

These formulas are used to perform calculations per row or group by Dimension. They can be used either within a Calculated Dimension, Calculated Measurement or in the Data Stream Mapping.

COUNT
Syntax: (groupKey)
Description: Calculates the row count. Optionally it can be calculated for any group aggregation provided by the groupKey parameter.
Example:
COUNT("csv['campaign']") will result in the total count of rows per campaign.

SUM
Syntax: (measurementName,measurementValue,groupKey)
Description: Computes the sum of the given measurement. Optionally it can be calculated for any group aggregation provided by the groupKey parameter. Make sure to select a different measurementName for each field that uses this formula in the Data Stream mapping, as the measurementName acts like a global variable within the Data Stream itself.
Example:
SUM('sumOfClicks', csv['clicks']) will result in the total sum for the clicks column.
SUM('sumOfImpressions',csv['impressions'],csv['campaign_key']) will result in the sum of impressions per campaign key.

Logical Formulas

AND
Syntax: (condition1, condition2)
Description: This formula is used to check if all conditions entered are true.
Example:
AND(csv['campaign_key'] contains '2018', csv['media_buy_key'] == 'ABC') will return true if both conditions are met.

IF

Syntax: (condition, resultIfTrue, resultIfFalse)
Description: Returns the second argument if the first argument is true, and the third argument if otherwise.
Example:
IF(csv['Media Buy Name'] contains '_DIG', 'Digital', 'Print')
Explanation: If the Media Buy Name field contains the substring '_DIG', return 'Digital', else return 'Print'.

NOT
Syntax: (condition)
Description: This formula reverses the logic of its argument, true becomes false and false becomes true.
Example:
NOT(csv['campaign_key'] contains '2018') will return true for all campaign key values that do NOT contain '2018'.

OR
Syntax: (condition1, condition2)
Description: This formula checks if either one of the conditions is true.
Example:
OR(csv['campaign_key'] contains '2018', csv['media_buy_key'] == 'ABC') will return true if any of the 2 conditions are met.

Type Conversion

COMPRESS
Syntax: (value1)
Description: Compresses a string using gzip and return it in base64.
The formula compresses a long string into shorter strings. When using this formula you will need to uncompress the data once it is uploaded into the platform, using the UNCOMPRESS Formula.
Note that when applied to short strings (roughly, below 300 characters), this formula will produce a longer encoded string. If the strings you are applying the formula to are not stable lengthwise, it is advised you use an IF formula instead.
Example:
COMPRESS(csv['section'])

UNCOMPRESS(value1)
Syntax: UNCOMPRESS(value1)
Description: This formula is used to uncompress strings encoded by the COMPRESS formula.
Example:
UNCOMPRESS(csv['section'])

ENCODE_BASE64
Syntax: ENCODE_BASE64(value1)
Description: This formula encodes a long string into shorter strings. When using this formula you will need to decode the data once it is uploaded into the platform, this is done using the DECODE_BASE64 Formula.

Note that when applied to short strings (roughly, below 300 characters), this formula will produce a longer encoded string. If the strings you are applying the formula to are not stable lengthwise it is advised you use an IF formula instead.
For more information about the ENCODE_BASE64 formula, see the following sites:
https://www.base64encode.org/ (https://www.base64encode.org/)
https://www.base64decode.org/ (https://www.base64decode.org/)
Example:
ENCODE_BASE64('abc123') = 'YWJjMTIz'
ENCODE_BASE64(csv['section'])

DECODE_BASE64
Syntax: DECODE_BASE64(value1)
Description: This formula is used to decode strings encoded by the ENCODE_BASE64 formula.
For more information about the ENCODE_BASE64 formula, see the following sites:
https://www.base64encode.org/ (https://www.base64encode.org/)
https://www.base64decode.org/ (https://www.base64decode.org/)
Example:
DECODE_BASE64('YWJjMTIz') = 'abc123'
DECODE_BASE64(csv['section'])

FORMATDATE
Syntax: (date, pattern)
Description: Returns the string representation of the given date according to the supplied pattern.
For example: 'yyyy-MM-dd' --> 2014-02-25, 'dd MMM-yyyy' --> 25 Feb-2014, etc.
Note: Date formulas do not allow the use of a '0' prefix in 08 & 09. Be sure to insert numbers without a '0' prefix, e.g. '8' and not '08'.
Example:
FORMATDATE(TODAY(),'yyyy') will return the current year (e.g. 2018).
FORMATDATE(csv['date'], 'yyyy') will return the year of each date that is found in the csv['date'] column.

FORMATDATE_DEFAULT
Syntax: (date)
Description: This formula transforms Datorama's default date format from 'EEE MMM dd HH:mm:ss Z yyyy' to 'yyyy-MM-dd HH:mm:ss.S'.
Example:
FORMATDATE_DEFAULT(TODAY()) will return '2018-12-16 00:00:00.0'

MD5
Syntax: (stringToHash)
Description: Returns an MD5 hash of the given string.
For more information about this formula, **click here** (https://passwordsgenerator.net/md5-hash-generator/).
Example:
MD5('abc') = '900150983CD24FB0D6963F7D28E17F72'

NUMBER
Syntax: (value1)
Description: Converts a string to a number. If the string contains more than 1 number, then it returns the first number from the string.
Example:
NUMBER(csv['clicks'])
NUMBER('Marry had 2 little lambs') = 2
NUMBER('Marry had 2 little lambs and 4 dogs') = 2
NUMBER('Marry had no little lambs and no dogs') = 0

PARSEDATE
Syntax: { "dateString", "pattern" },
Description: Returns the date object that is specified by the given dateString using the given pattern.
For example: '2014-02-25' ---> 'yyyy-MM-dd', '25 Feb-2014' ---> 'dd MMM-yyyy'
Example:
PARSEDATE(csv['date'], 'yyyy-MM-dd')

SHA256
Syntax: (stringToHash)
Description: Returns a SHA256 hash of the given string.
Click here (https://passwordsgenerator.net/sha256-hash-generator/) for reference
Example:
SHA256('a') = CA978112CA1BBDCAFAC231B39A23DC4DA786EFF8147C4E72B9807785AFEE48BB
SHA256('abc') = BA7816BF8F01CFEA414140DE5DAE2223B00361A396177A9CB410FF61F20015AD
SHA256(csv['campaign_key'])

STRIP_HTML
Syntax: (value1)
Description: Strips html tags from a string.
Example:
STRIP_HTML(csv['page_path'])

Operators

a != b
Description: Checks if the values on both sides of the operator are not equal.
Example:
'foo' != 'bar' will return true
1 != 2 will return true
3 != 3 will return false

a % b
Description: Divides the number on the left by the number on the right and returns the remainder.
Example:

5 % 3 = 2
10 % 3 = 1
100 % 23 = 8

a && b
Description: Checks to see that the values on both sides of the operator are true.
Example:
csv['media _buy_key'] contains '2018' && csv['campaign_key'] == 'ABC'

a * b
Description: Multiplies the number on the left by the number on the right.
Example:
2 * 2 = 4
3 * 5 = 15
5 * 20 = 100

a + b
Description: In case the operands are numbers, the formula adds the value on the left to the value on the right. In case the operands are strings, the formula concatenates the string on the left to the string on the right.
Example:
1 + 2 = 3
'1' + '2' = '12'
'A' + 'B' = 'AB'

a – b
Description: Subtracts the value on the right from the value on the left.
Example:
5 - 3 = 2
10 - 4 = 6

a / b
Description: Divides the number on the left by the number on the right.
Example:
10 / 5 = 2
20 / 4 = 5

a < b
Description: Checks whether the value on the left side of the operator is less than the value on the right.
Example:
1 < 2 will return true
'a' < 'b' will return true
6 < 4 will return false
'z' < 'b' will return false
'b' < 'b' will return false

a <= b
Description: Checks whether the value on the left-hand side is less than or equal to the value on the right. The equal sign must be on the right side (e.g. =< will result in an error).

Example:
3 <= 6 will return true
4 <= 4 will return true
5 <= 1 will return false

a == b
Description: Checks whether the values on both sides of the operator are equal. For example, 'foo' == 'foo' is true.
Example:
1 == 1 will return true
'abc' == 'abc' will return true
'ABC' == 'abc' will return false

a > b
Description: Checks whether the value on the left-hand side of the operator is greater than the value on the right.
Example:
2 > 1 will return true
'b' > 'a' will return true
4 > 6 will return false
'b' > 'z' will return false
'b' > 'b' will return false

a >= b
Description: Checks whether the value on the left-hand side is greater than or equal to the value on the right.
Example:
6 >= 3 will return true
4 >= 4 will return true
1 >= 5 will return false

a || b
Description: Checks if either the value on the left or the right is true.
Example:
csv['media_buy_key'] == 'ABC' || csv['campaign_key'] contains '2018'
Will return true if any of the 2 conditions are met.

Java Based IF-Else
Description: Returns the second argument if the first argument is true, and the third argument if otherwise.
Syntax:
```
if(Boolean_expression){
//Executes when the Boolean expression is true
}else{
//Executes when the Boolean expression is false
}
```
Example:
```
if([Site_Name] contains 'Twit'){
'Twitter';
} else {
'Other;
}
```

Explanation: If the Site Name contains the substring 'Twit', returns a value called 'Twitter'. If the Site Name does not contain the substring 'Twit', returns a value called 'Other'.