String Functions

Tableau Desktop Reference Guide

String functions manipulate string data. String functions are useful for harvesting meaningful information found within strings and for improving the presentation of your data.

Parse out portions of a string using functions such as **FIND** and **FINDNTH** to find the position of delimiters or spaces and then using the **LEFT**, **RIGHT**, or **MID** functions to return or replace parts of the given string. Alternatively, use the **SPLIT** function when possible.

Some string functions use an optional starting position argument so the action is performed starting at the position given. The first character in the string is position 1. Other string functions use an optional length argument. If the length is not given, the action is performed for the entirety of the string.

Pattern Matching

Functions such as a **REGEXP_REPLACE** work with a regular expression, also called a rational expression, and can be used for pattern matching, similar to a "find and replace" command. This pattern matching syntax conforms to the International Components for Unicode (ICU) standard, and ranges from precise equality to general similarity using wildcard characters.

These functions are useful when dealing with strings that follow a particular pattern, such as phone numbers, email names, or part numbers. You can use these functions to isolate, replace, or test the validity of parts of these strings.

Examples

These are some examples of common string functions:

Function Syntax	Purpose	Example
LEFT(string, number) **RIGHT** and **MID** are similar	Returns the leftmost number of characters in a string.	**LEFT**(**[Postal Code]**, 2) returns "98" if the **[Postal Code]** is "98103"
SPLIT(string, delimiter, token number)	Returns a substring from a string, using a delimiter to divide the string into substrings and then using the token to determine which substring to return.	**SPLIT**(**[Phone]**, "-", 2) returns "633" if the value of **[Phone]** is "206-633-3400" since "633" is the 2nd token substring delimited by a hyphen.
CONTAINS(string, substring) **ENDSWITH** and **STARTSWITH** are similar	Returns true if the given string contains the specified substring, otherwise returns false.	**CONTAINS**(**[Product Name]**, "Phone") returns true if **[Product Name]** is "Apple iPhone" and false if **[Product Name]** is "Apple iPad"
FIND(string, substring, [start]) **FINDNTH** is similar	Returns the position of the substring within the given string, or 0 if the substring is not found.	**FIND**(**[Customer Name]**, ",") returns 4 if the **[Customer Name]** is "Doe, Jane" since that is the position of the comma delimiter.

MIN(expression) or **MIN** (expr1, expr2) **MAX** is similar	Returns the minimum of a single expression across all records, or the minimum of the two expressions for each record. For strings, the result is based on alphabetical order.	**MIN**(**[Vendor Name]**) returns "AAA Towing" if that is the first alphabetical value for **[Vendor Name]**.
REGEXP_REPLACE(string, pattern, replacement string) **REGEXP_EXTRACT**, **REGEXP_EXTRACTNTH**, and **REGEXP_MATCH** are similar.	Returns a string where the matching pattern is substituted with the replacement string.	**REGEXP_REPLACE**(**[Part ID]**, "\s", "-") returns "abc-123" if the **[Part ID]** is "abc 123" since the "\s" matches a white space character, which will be replaced by a hyphen, "-".

Many other common string functions are available and include:

Function	Purpose
REPLACE, **SPACE**	Return or replace parts of the given string.
ASCII, **CHAR**	Convert between ASCII codes and characters.
LOWER, **UPPER**	Update the string to upper or lowercase characters.
ISDATE	Return true if the string is a valid date.

Number Functions

Tableau Desktop Reference Guide

Number functions perform mathematical functions on numbers and return the result as another number. Number functions are useful when a simple math function was not already performed in the underlying data.

When you join or blend data sources, you can have some records with no value (null) in the shared field. You cannot do math with null values, but you can use the ZN function to replace null values with zero (0), which will then allow you to perform any required math functions.

Examples

These are some examples of common number functions:

Function Syntax	Purpose	Example
ABS(number)	Returns the absolute value of a number.	**ABS**(**[Profit Variance]**) returns 10 if the [**Profit Variance]** value is 10 or -10.
ROUND(number, **[decimals]**) **CEILING** and **FLOOR** are similar	Rounds the number to the number of decimal points specified or to the next whole number if none are specified.	**ROUND**(**[Test Score]**) returns 92 if the **[Test Score]** is between 91.5 and 92.4
MIN(expression) or **MIN**(expr1, expr2) **MAX** is similar	Returns the minimum of a single expression across all records, or the minimum of the two expressions for each record.	**MIN**(**[Quantity]**) returns 42 if that is the smallest value for **[Quantity]**.
ZN(expression)	Returns the expression if it is not null, otherwise it returns zero.	**ZN**(**[Return Quantity]**) is 2 if the value of **[Return Quantity]** is equal to 2, and 0 if **[Return Quantity]** is null.

Many other common mathematical functions are available and include:

Function	Purpose
DIV, **SIGN**, **SQRT**, **SQUARE**	Basic math functions
ACOS, **ASIN**, **ATAN**, **ATAN2**, **COS**, **COT**, **SIN**, **TAN**	Trigonometric functions
DEGREES, **RADIANS**	Angular conversion
HEXBINX, **HEXBINY**	Mapping
EXP, **LN**, **LOG**, **POWER**	Power and logarithmic functions
PI	Math constant

Logical Functions

Tableau Desktop Reference Guide

Use logical functions to evaluate conditions and return values based on the result. Logical functions are useful to create specific views by categorizing, filtering/excluding, or combining records. Logical functions using comparison operators can be used to create dynamic labels, spotlight formatting, or create virtual bins to categorize data, similar to how KPIs (Key Performance Indicators) are evaluated.

The **IF/THEN/ELSE/END** function is used for evaluating test conditions and returning values based on the results. **IF** statements can also be extended using **ELSEIF** tests. The **IIF** function is a shortcut way of writing an **IF** function.

The **CASE** function can be useful when searching for a match to an expression, but also useful when an **IF** statement would be very complex, due to many test conditions.

The **ISDATE** function can be used to test if a string is a valid date.

Testing for Null

There are several functions that test for null results, **IFNULL**, **ISNULL**, and **ZN** that are useful when your data has unknown or no values for some fields. **ZN** returns a value of zero (0) if the expression is null.

Examples

These are examples of how logical functions can be used:

Function Syntax	Purpose	Example
IF test **THEN** value1 [**ELSE** value2] **END** **ELSEIF** can also be used **IIF** is similar	Evaluates a sequence of test conditions and returns the first value for the condition that is true.	**IF [Cost]** > **[Budget]** **THEN** "Over" **ELSE** "Under" **END** **This** returns the string "Over" if the value of [Cost] is $7 and [Budget] is equal to $5, but will return the string "Under" if the value of [Cost] is $7 and the [Budget] is equal to $9.
CASE expression **WHEN** value1 **THEN** return1 **WHEN** value2 **THEN** return2 [**ELSE** else value] **END**	Search for a match to an expression and return a value based on the match. If there is no match, the optional else value is returned. If there is no default, then null is returned.	**CASE [Country]** **WHEN** "United States" **THEN** "USA" **WHEN** "Canada" **THEN** "CAN" **ELSE** "Other" **END** **This** returns "Other" if **Country** is "Mexico".

IFNULL(expression1, expression2) **ISNULL** and **ZN** are similar	Returns the first expression if it is not null, otherwise it returns the second expression.	**IFNULL**(**[Grade],** "not tested") returns "A" if the value of **[Grade]** is equal to "A", and returns "not tested" if the value of **[Grade]** is null.

Tableau eLearning

Primary Functions

Tableau Desktop Reference Guide

The primary functions perform computations on the table.

Note that since the calculations are computed on the aggregated table data, field arguments for these functions must be aggregated as well, for example **SUM**(**[Profit]**).

Function Syntax	Purpose	Example
TOTAL (expression)	Returns the total for the given expression in the current partition.	**TOTAL**(**SUM**(**[Sales]**)) returns the total for the sum of sales based on the current scope and direction.
LOOKUP (expression,[offset])	Returns the value of the expression in a target row, specified as a relative offset from the current row. If the offset is –1, then the result will be returned for the previous value in the scope and direction.	**LOOKUP**(**SUM**(**[Profit]**), **FIRST**()+2) computes the **SUM**(**[Profit]**) in the third row of the partition.
PREVIOUS_VALUE (expression)	Returns the value of the expression in the previous row.	**SUM**(**[Profit]**) + **PREVIOUS_VALUE**(1) computes the running total of **SUM**(**[Profit]**).
RUNNING_SUM (expression) **RUNNING_AVG**, **RUNNING_MAX**, **RUNNING_MIN**, and **RUNNING_COUNT** are similar	Returns the running sum of the given expression, from the first row in the partition to the current row.	**RUNNING_SUM**(**SUM**(**[Profit]**)) computes the running sum of **SUM**(**[Profit]**)
WINDOW_AVG(expression, [start, end]) **WINDOW_SUM**, **WINDOW_MAX**, **WINDOW_MIN**, **WINDOW_MEDIAN**, **WINDOW_COUNT**, **WINDOW_PERCENTILE**, **WINDOW_STDEV**, **WINDOW_STDEVP**, **WINDOW_VAR**, **WINDOW_VARP** are all similar	Returns the average of the expression within the window. If the optional start and end are omitted, the entire partition is used.	**WINDOW_AVG**(**SUM**(**[Profit]**), **FIRST**()+1, 0) computes the average of **SUM**(**[Profit]**) from the second row to the current row.l

Function Syntax	Purpose	Example
WINDOW_CORR (expression1, expression2, [start, end])	Returns the Pearson correlation coefficient of the two expressions within the window. If the optional start and end are omitted, the entire partition is used.	**WINDOW_CORR**(**SUM**(**[Sales]**), **SUM**(**[Profit]**)) returns a value from – 1 to 1. The result is equal to 1 for an exact positive linear relationship, 0 for no linear relationship, and –1 for an exact negative linear relationship.
WINDOW_COVAR (expression1, expression2, [start, end])	**WINDOW_COVARP** is similar, but for a population, instead of a sample. Returns the sample covariance of two expressions within the window. If the optional start and end are omitted, the entire partition is used. If the two expressions are the same, a value is returned that indicates how widely the variables are distributed.	**WINDOW_COVAR**(**SUM**(**[Sales]**), **SUM**(**[Profit]**)) returns a positive number if the expressions tend to vary together, on average.
RANK (expression, [order]) **RANK_DENSE**, **RANK_**. **MODIFIED**, **RANK_ UNIQUE**, **RANK_**, **PERCENTILE**	Returns the standard competition rank for the current row in the partition.	**RANK**(**AVG**(**[Test Score]**))

In the **WINDOW** functions, the window is defined by means of offsets from the current row. Use the helper functions **FIRST** ()+n and **LAST**()-n for offsets from the first or last row in the partition. If the start and end are omitted, the entire partition is used.

To calculate a population and sample standard deviation and variance, use the **WINDOW_STDEV**,

WINDOW_STDEVP, **WINDOW_VAR**, and **WINDOW_VARP** functions.

To calculate the measure or extent of joint variability of two expressions within a window, use the **WINDOW_CORR**, **WINDOW_COVAR**, and **WINDOW_ COVARP** functions.

In the RANK functions you can optionally use 'asc' or 'desc' to specify the ranking order. The default is descending.

Nulls are ignored in ranking functions. They are not numbered and they do not count against the total number of records in percentile rank calculations.

The **RANK** functions vary on how they process identical values, such as when the ranks for the set of values (6, 9, 9, 14) are computed:

- **RANK**: Identical values are assigned an identical rank, for example 1, 2, 2, 4.
- **RANK_DENSE**: Identical values are assigned an identical rank, but no gaps are inserted into the number sequence, for example 1, 2, 2, 3.

Script Table Calculation Functions

Additional functions are available to interact directly with external service scripts, including those in R, MATLAB, and Python. The **SCRIPT** functions vary in the data type returned, Boolean, string, integer, or real number: **SCRIPT_BOOL**, **SCRIPT_STR**, **SCRIPT_INT**, and **SCRIPT_REAL**.

Tableau eLearning

Helper Functions

Tableau Desktop Reference Guide

Use the helper functions to provide information about the relative position or size of an item within the current scope and direction.

The results from **FIRST**, **LAST**, **INDEX**, and **SIZE** functions can be used as input to other calculations or functions, such as the **LOOKUP** function. For example, use **FIRST**() + n and **LAST**() – n as part of your offset definition for a target relative to the first or last rows in the partition.

Function Syntax	Purpose	Example
INDEX()	Returns the position of the current value, given the chosen scope and direction.	**INDEX**() returns the index for the current row. The first row index starts at 1.
FIRST() **LAST**() is similar	Returns the offset of the position of the current value.	**FIRST**() returns 0 if the current value is in the first row and returns –1 if the current value is in the second row, and so on. **LAST**() returns 0 if the current value is in the last row and returns 1 for the previous row and so on.
SIZE()	Returns the number of rows in the current scope.	**SIZE**() returns 12 if the current part of the table shows sales for months in a year.

FAQ

These functions will return numbers with decimals, so change the format to standard if you want to show these as stand–alone calculated fields.Note that these functions are followed by parentheses, even though they are empty.For more information about helper functions, check out the **Help topic**.

Date Functions

Tableau Desktop Reference Guide

Date functions operate on date fields in order to manipulate, compare, or query information about them. If you want to refer to a date as a literal, that is input as is, surround the input date with the pound symbol, (for example: #December 20, 2016#).

Some functions use a start_of_week option which defaults based on the data source value. The syntax for date format parameters follows the International Components for Unicode (ICU) definitions.

Date Parts

Many date functions use date_part, which indicates the specific part of the date you would like to manipulate or query. Note that the date_part argument is case sensitive and must be all in lowercase. Valid date_part values are:

date _part	Values
'year'	Four-digit year
'quarter'	1-4
'month'	1-12 or "January", "February", and so on
'dayofyear'	Day of the year; Jan 1 is 1, Feb 1 is 32, and so on
'day'	1-31
'weekday'	1-7 or "Sunday", "Monday", and so on
'week'	1-52
'hour'	0-23
'minute'	0-59
'second'	0-60

Examples

These are some examples of common date functions:

Function Syntax	Purpose	Example
MONTH(date) **YEAR** and **DAY** are similar **DATEPART**(date_part, date, [start_of_week]) is also similar	Returns a number representing the year (or date_part) of a given date.	**MONTH**(**[Ship Date]**) returns 12 if the **[Ship Date]** is equal to #December 20, 2016#.

DATENAME(date_part, date, **[start_of_week]**)	Similar to **DATEPART**, but returns the date_part of the given date as a string instead of as a number.	**DATENAME**('month', **[Ship Date]**) returns "December" if the **[Ship Date]** is equal to #2016-12-20#.
DATEPARSE(format, string)	Works in the reverse of **DATENAME** by converting a string into a date/time using the given ICU format.	**DATEPARSE**("dd.MMMM.yyyy", **[Delivered On]**) returns #10/28/2015# if **[Delivered On]** is equal to the string "28.October.2015".
DATEDIFF(date_part, start_date, end_date, [start_of_week])	Returns the difference between the two dates using the units of date_ part.	**DATEDIFF**('month', **[Order Date]**, #2016-01-01#) returns 1 if the **[Order Date]** is #February 23, 2016# and -12 if the **[Order Date]** is #January 3, 2015#.
DATEADD(date_part, increment, date)	Returns a date in which the increment has been added to the given date. The type of increment is specified using the date_part	**DATEADD**('month', 6, **[Order Date])** returns #July 1, 2016# if **[Order Date]** is #January 1, 2016#.
DATETRUNC(date_part, date, [start_of_week])	Truncates or "rounds down" the given date to the level of the specified date_part.	**DATETRUNC**('month', **[Order Date]**) returns #February 1, 2005# if **[Order Date]** is #February 17, 2005#. DATETRUNC('quarter', **[Order Date]**) returns #April 1, 2015# if **[Order Date]** is #June 21, 2015#.
MIN(expression) or **MIN**(expr1, expr2) **MAX** is similar	Returns the earliest date across all records, or the earlier of the two dates for each record.	**MIN**(**[Order Date]**) returns $2012-01-03# if the oldest order date is #January 3, 2012#.

Other date functions are available and include:

Function	Purpose
TODAY, **NOW**	Return the current date or current date and time.
ISDATE	Check if a given string is a valid date.
MAKEDATE, **MAKETIME**, **MAKEDATETIME**	Returns a date, time, or datetime value constructed from the arguments given.

Performance Considerations Related to Date Functions

To improve the performance of your worksheet, take these considerations into account when using date functions:

Use the **NOW** function only if you need the time stamp level of detail, otherwise, use **TODAY** for date level calculations.

If possible, use the **DATETRUNC**, **DATEADD**, and **DATEDIFF** functions instead of more complex tests using multiple date functions, such as **YEAR** and **MONTH**.

Use **DATEPARSE** if your data has dates that are stored as strings or numeric time stamps (that is, not stored in native date formats).

Aggregate Functions

Tableau Desktop Reference Guide

The aggregate functions allow you to summarize data, for example by calculating the average of numbers in a measure. Aggregate functions perform a calculation on a set of values, at the level of granularity of the view, and return a single value. An aggregated calculation appears with the letters **AGG** in front of it when placed on a shelf or on the **Marks** card.

To calculate a population and sample standard deviation and variance, use the **STDEV**, **STDEVP**, **VAR**, and **VARP** functions.

To calculate the measure or extent of joint variability of two expressions, use the **CORR**, **COVAR**, and **COVARP** functions.

NOTE The expressions used in any function must either all be aggregated or all disaggregated. The **MAX**, **MIN**, and **ATTR** functions allow you to have all expressions in a function use a similar aggregation level.

Examples

These are examples of the aggregate functions:

Function Syntax	Purpose	Example
AVG(expression) **SUM**, **MEDIAN**, **PERCENTILE**, **STDEV**, **STDEVP**, **VAR**, **VARP** are similar	Returns the average of all the values in the expression.	**AVG**(**[Sales]**) returns $500 if the average of the **[Sales]** values is $500.
COUNT(expression)	Returns the number of items in a group. Nulls are not counted.	**COUNT**(**[Vendor ID]**) will return 123, if there are 123 records that have non-null **[Vendor ID]** field values.
COUNTD(expression)	Returns the number of unique items in a group.	**COUNTD**(**[Vendor ID]**)will return 50 if there are only 50 records that have unique **[Vendor ID]** field values.
MAX(expression)		
MIN is similar	Returns the maximum of an expression across all records.	**MAX**(**[Sales]**) returns $1,000 if $1,000 is the largest number in **[Sales]**.
ATTR(expression)	Returns the value of the expression if it has a single value for all rows, otherwise it returns an asterisk (*). This allows you to check if values are the same.	**ATTR**(**[Segment]**) returns 'Corporate' if the value of **[Segment]** is 'Corporate' at the level of granularity in the view, otherwise it returns '*'.

CORR(expression1, expression2)	Returns the Pearson correlation coefficient of the two expressions.	**CORR**(**[Sales]**, **[Profit]**) returns a value from –1 to 1. The result is equal to 1 for an exact positive linear relationship, 0 for no linear relationship, and –1 for an exact negative linear relationship.
COVAR(expression1, expression2) **COVARP** is similar, but for a population, instead of a sample	Returns the sample covariance of two expressions. If the two expressions are the same, a value is returned that indicates how widely the variables are distributed.	**COVAR**(**[Sales]**, **[Profit]**) returns a positive number if the expressions tend to vary together, on average.

Tableau eLearning

Helper Functions

Tableau Desktop Reference Guide

Use the helper functions to provide information about the relative position or size of an item within the current scope and direction.

The results from **FIRST**, **LAST**, **INDEX**, and **SIZE** functions can be used as input to other calculations or functions, such as the **LOOKUP** function. For example, use **FIRST**() + n and **LAST**() – n as part of your offset definition for a target relative to the first or last rows in the partition.

Function Syntax	Purpose	Example
INDEX()	Returns the position of the current value, given the chosen scope and direction.	**INDEX**() returns the index for the current row. The first row index starts at 1.
FIRST() **LAST**() is similar	Returns the offset of the position of the current value.	**FIRST**() returns 0 if the current value is in the first row and returns –1 if the current value is in the second row, and so on. **LAST**() returns 0 if the current value is in the last row and returns 1 for the previous row and so on.
SIZE()	Returns the number of rows in the current scope.	**SIZE**() returns 12 if the current part of the table shows sales for months in a year.

FAQ

These functions will return numbers with decimals, so change the format to standard if you want to show these as stand–alone calculated fields.Note that these functions are followed by parentheses, even though they are empty.For more information about helper functions, check out the **Help topic**.

Logical Functions

Tableau Desktop Reference Guide

Use logical functions to evaluate conditions and return values based on the result. Logical functions are useful to create specific views by categorizing, filtering/excluding, or combining records. Logical functions using comparison operators can be used to create dynamic labels, spotlight formatting, or create virtual bins to categorize data, similar to how KPIs (Key Performance Indicators) are evaluated.

The **IF/THEN/ELSE/END** function is used for evaluating test conditions and returning values based on the results. **IF** statements can also be extended using **ELSEIF** tests. The **IIF** function is a shortcut way of writing an **IF** function.

The **CASE** function can be useful when searching for a match to an expression, but also useful when an **IF** statement would be very complex, due to many test conditions.

The **ISDATE** function can be used to test if a string is a valid date.

Testing for Null

There are several functions that test for null results, **IFNULL**, **ISNULL**, and **ZN** that are useful when your data has unknown or no values for some fields. **ZN** returns a value of zero (0) if the expression is null.

Examples

These are examples of how logical functions can be used:

Function Syntax	Purpose	Example
IF test **THEN** value1 [**ELSE** value2] **END** **ELSEIF** can also be used **IIF** is similar	Evaluates a sequence of test conditions and returns the first value for the condition that is true.	**IF [Cost]** > **[Budget]** **THEN** "Over" **ELSE** "Under" **END** **This** returns the string "Over" if the value of [Cost] is $7 and [Budget] is equal to $5, but will return the string "Under" if the value of [Cost] is $7 and the [Budget] is equal to $9.
CASE expression **WHEN** value1 **THEN** return1 **WHEN** value2 **THEN** return2 [**ELSE** else value] **END**	Search for a match to an expression and return a value based on the match. If there is no match, the optional else value is returned. If there is no default, then null is returned.	**CASE [Country]** **WHEN** "United States" **THEN** "USA" **WHEN** "Canada" **THEN** "CAN" **ELSE** "Other" **END** **This** returns "Other" if **Country** is "Mexico".

IFNULL(expression1, expression2) **ISNULL** and **ZN** are similar	Returns the first expression if it is not null, otherwise it returns the second expression.	**IFNULL**(**[Grade],** "not tested") returns "A" if the value of **[Grade]** is equal to "A", and returns "not tested" if the value of **[Grade]** is null.

Type Conversion Functions

Tableau Desktop Reference Guide

All fields in a data source have a data type, such as date, datetime, string, integer, or floating point number. You can use type conversion functions to convert the result of any expression from one data type to another. This is useful in order to use the result in a calculation that requires a different data type as input.

Examples

These are some examples of he type conversion functions:

Function Syntax	Purpose	Example
DATE(expression) **DATETIME**, **MAKEDATE**, **MAKEDATETIME**, and **MAKETIME** are also similar	Returns a date when given a number, string, or date expression.	**DATE**(**[Ship Date]**) returns #November 15, 1998# if **[Ship Date]** is a string with the value "November 15, 1998".
STR(expression)	Returns a string from the given expression.	**STR**(**[Age]**) returns "17" if the value of **[Age]** is the integer 17.
INT(expression)	Returns an integer by truncating the given expression. If the number given is negative, the function will return the integer closer to zero.	**INT**(**[Partial Order]**) returns 1 if **[Partial Order]** is 1.5.
INT([Profit]) returns -2 if [Profit] is -2.8.		
FLOAT(expression)	Returns a floating point number from the given expression.	**FLOAT**(**[Age]**) returns 17.000 if the value of **[Age]** is the integer 17.