make_datetime()

Article • 02/15/2023

Creates a datetime scalar value between the specified date and time.

Syntax

`make_datetime(`*year*, *month*, *day*`)`

`make_datetime(`*year*, *month*, *day*, *hour*, *minute*`)`

`make_datetime(`*year*, *month*, *day*, *hour*, *minute*, *second*`)`

Parameters

Name	Type	Required	Description
year	int	✓	The year value between 0 to 9999.
month	int	✓	The month value between 1 to 12.
day	int	✓	The day value between 1 to 28-31, depending on the month.
hour	int		The hour value between 0 to 23.
minute	int		The minute value between 0 to 59.
second	double		The second value between 0 to 59.9999999.

Returns

If successful, the result will be a datetime value, otherwise, the result will be null.

Example

Run the query

Kusto

```
print year_month_day = make_datetime(2017,10,01)
```

Output

year_month_day
2017-10-01 00:00:00.0000000

Run the query

```
Kusto
print year_month_day_hour_minute = make_datetime(2017,10,01,12,10)
```

Output

year_month_day_hour_minute
2017-10-01 12:10:00.0000000

Run the query

```
Kusto
print year_month_day_hour_minute_second =
make_datetime(2017,10,01,12,11,0.1234567)
```

Output

year_month_day_hour_minute_second
2017-10-01 12:11:00.1234567

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

make_timespan()

Article • 01/26/2023

Creates a timespan scalar value from the specified time period.

Syntax

`make_timespan(`*hour*, *minute*`)`

`make_timespan(`*hour*, *minute*, *second*`)`

`make_timespan(`*day*, *hour*, *minute*, *second*`)`

Parameters

Name	Type	Required	Description
day	int	✓	The day.
hour	int	✓	The hour. A value from 0-23.
minute	int		The minute. A value from 0-59.
second	real		The second. A value from 0 to 59.9999999.

Returns

If the creation is successful, the result will be a timespan value. Otherwise, the result will be null.

Example

Run the query

Kusto

```
print ['timespan'] = make_timespan(1,12,30,55.123)
```

Output

timespan
1.12:30:55.1230000

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

max_of()

Article • 01/26/2023

Returns the maximum value of all argument expressions.

Syntax

`max_of(`*arg*`,` *arg_2*`,` [*arg_3*`,` ...]`)`

Parameters

Name	Type	Required	Description
arg_i	scalar	✓	The values to compare.

- All arguments must be of the same type.
- Maximum of 64 arguments is supported.
- Non-null values take precedence to null values.

Returns

The maximum value of all argument expressions.

Examples

Find the largest number

Run the query

Kusto

```
print result = max_of(10, 1, -3, 17)
```

Output

result
17

Find the maximum value in a data-table

Notice that non-null values take precedence over null values.

Run the query

Kusto

```
datatable (A: int, B: int)
[
    1, 6,
    8, 1,
    int(null), 2,
    1, int(null),
    int(null), int(null)
]
| project max_of(A, B)
```

Output

result
6
8
2
1
(null)

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

merge_tdigest()

Article • 06/12/2023

Merges `tdigest` results (scalar version of the aggregate version tdigest_merge()).

Read more about the underlying algorithm (T-Digest) and the estimated error here.

> The `merge_tdigest()` and `tdigest_merge()` functions are equivalent

> ⓘ **Important**
>
> The results of tdigest() and tdigest_merge() can be stored and later retrieved. For example, you may want to create daily percentiles summary, which can then be used to calculate weekly percentiles. However, the precise binary representation of these results may change over time. There's no guarantee that these functions will produce identical results for identical inputs, and therefore we don't advise relying on them.

Syntax

`merge_tdigest(`*exprs*`)`

Parameters

Name	Type	Required	Description
exprs	dynamic	✓	One or more comma-separated column references that have the `tdigest` values to be merged.

Returns

The result for merging the columns `*Expr1*`, `*Expr2*`, ... `*ExprN*` to one `tdigest`.

Example

Run the query

Kusto

```
range x from 1 to 10 step 1
| extend y = x + 10
| summarize tdigestX = tdigest(x), tdigestY = tdigest(y)
| project merged = merge_tdigest(tdigestX, tdigestY)
| project percentile_tdigest(merged, 100, typeof(long))
```

Output

percentile_tdigest_merged
20

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

min_of()

Article • 01/26/2023

Returns the minimum value of several evaluated scalar expressions.

Syntax

`min_of` `(`*arg*`,` *arg_2*`,` [*arg_3*, ...]`)`

Parameters

Name	Type	Required	Description
arg, *arg_2*, ...	scalar	✓	A comma separated list of 2-64 scalar expressions to compare. The function returns the minimum value among these expressions.

- All arguments must be of the same type.
- Maximum of 64 arguments is supported.
- Non-null values take precedence to null values.

Returns

The minimum value of all argument expressions.

Examples

Find the maximum value in an array:

Run the query

Kusto

```
print result=min_of(10, 1, -3, 17)
```

Output

result
-3

Find the minimum value in a data-table. Non-null values take precedence over null values:

Run the query

Kusto

```
datatable (A: int, B: int)
[
    5, 2,
    10, 1,
    int(null), 3,
    1, int(null),
    int(null), int(null)
]
| project min_of(A, B)
```

Output

result
2
1
3
1
(null)

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

monthofyear()

Article • 01/26/2023

Returns the integer number from 1-12 representing the month number of the given year.

> The `monthofyear()` and `getmonth()` functions are equivalent

Syntax

`monthofyear(`*date*`)`

Parameters

Name	Type	Required	Description
date	datetime	✓	The date for which to find the month number.

Returns

An integer from 1-12 representing the month number of the given year.

Example

Run the query

Kusto

```
print result=monthofyear(datetime("2015-12-14"))
```

Output

result
12

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

new_guid()

Article • 01/26/2023

Returns a random GUID (Globally Unique Identifier).

Syntax

`new_guid()`

Returns

A new value of type guid.

Example

Run the query

```
Kusto

print guid=new_guid()
```

Output

guid
2157828f-e871-479a-9d1c-17ffde915095

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

not()

Article • 03/12/2023

Reverses the value of its `bool` argument.

Syntax

`not(` *expr* `)`

Parameters

Name	Type	Required	Description
expr	scalar	✓	An expression that evaluates to a boolean value. The result of this expression will be reversed.

Returns

Returns the reversed logical value of its `bool` argument.

Example

> Run the query

Kusto

```
print not(false) == true
```

Output

print_0
true

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

now()

Article • 06/19/2023

Returns the current UTC time, optionally offset by a given timespan.

The current UTC time will stay the same across all uses of `now()` in a single query statement, even if there's technically a small time difference between when each `now()` runs.

Syntax

`now(` [*offset*] `)`

Parameters

Name	Type	Required	Description
offset	timespan		A timespan to add to the current UTC clock time. The default value is 0.

Returns

The current UTC clock time, plus the *offset* time if provided, as a `datetime`.

Examples

Find time elapsed from a given event

The following example shows the time elapsed since the start of the storm events.

Run the query

Kusto

```
StormEvents
| extend Elapsed=now() - StartTime
| take 10
```

Get the date relative to a specific time interval

Run the query

Kusto

```
let T = datatable(label: string, timespanValue: timespan) [
    "minute", 60s,
    "hour", 1h,
    "day", 1d,
    "year", 365d
];
T
| extend timeAgo = now() - timespanValue
```

Output

label	timespanValue	timeAgo
year	365.00:00:00	2022-06-19T08:22:54.6623324Z
day	1.00:00:00	2023-06-18T08:22:54.6623324Z
hour	01:00:00	2023-06-19T07:22:54.6623324Z
minute	00:01:00	2023-06-19T08:21:54.6623324Z

> **Note**
>
> This operation can be accomplished with the **ago() function**.

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

pack_all()

Article • 01/10/2023

Creates a dynamic property bag object from all the columns of the tabular expression.

> ⓘ Note
>
> The representation of the returned object isn't guaranteed to be byte-level-compatible between runs. For example, properties that appear in the bag may appear in a different order.

Syntax

`pack_all(`[*ignore_null_empty*]`)`

Parameters

Name	Type	Required	Description
ignore_null_empty	bool		Indicates whether to ignore null/empty columns and exclude them from the resulting property bag. The default value is `false`.

Example

The following query will use `pack_all()` to create columns for the below table.

SourceNumber	TargetNumber	CharsCount
555-555-1234	555-555-1212	46
555-555-1234	555-555-1213	50
555-555-1313		42
	555-555-3456	74

Run the query

Kusto

```
datatable(SourceNumber:string,TargetNumber:string,CharsCount:long)
[
'555-555-1234','555-555-1212',46,
'555-555-1234','555-555-1213',50,
'555-555-1313','',42,
'','555-555-3456',74
]
| extend Packed=pack_all(), PackedIgnoreNullEmpty=pack_all(true)
```

Output

SourceNumber	TargetNumber	CharsCount	Packed	PackedIgnoreNullEmpty
555-555-1234	555-555-1212	46	{"SourceNumber":"555-555-1234", "TargetNumber":"555-555-1212", "CharsCount": 46}	{"SourceNumber":"555-555-1234", "TargetNumber":"555-555-1212", "CharsCount": 46}
555-555-1234	555-555-1213	50	{"SourceNumber":"555-555-1234", "TargetNumber":"555-555-1213", "CharsCount": 50}	{"SourceNumber":"555-555-1234", "TargetNumber":"555-555-1213", "CharsCount": 50}
555-555-1313		42	{"SourceNumber":"555-555-1313", "TargetNumber":"", "CharsCount": 42}	{"SourceNumber":"555-555-1313", "CharsCount": 42}
	555-555-3456	74	{"SourceNumber":"", "TargetNumber":"555-555-3456", "CharsCount": 74}	{"TargetNumber":"555-555-3456", "CharsCount": 74}

> ⓘ Note
>
> There is a difference between the *Packed* and the *PackedIgnoreNullEmpty* columns in the last two rows of the above example. These two rows included empty values that were ignored by *pack_all(true)*.

Feedback

Was this page helpful? Yes No

pack_array()

Article • 01/10/2023

Packs all input values into a dynamic array.

Syntax

`pack_array(`*value1*`,` [*value2*, ...]`)`

`pack_array(*)`

Parameters

Name	Type	Required	Description
value1...valueN	string	✓	Input expressions to be packed into a dynamic array.
The wildcard `*`	string		Providing the wildcard `*` will pack all input columns into a dynamic array.

Returns

A dynamic array that includes the values of *value1*, *value2*, ... *valueN*.

Example

Run the query

Kusto

```
range x from 1 to 3 step 1
| extend y = x * 2
| extend z = y * 2
| project pack_array(x, y, z)
```

Output

Column1
[1,2,4]

Column1
[2,4,8]
[3,6,12]

Run the query

Kusto

```
range x from 1 to 3 step 1
| extend y = tostring(x * 2)
| extend z = (x * 2) * 1s
| project pack_array(x, y, z)
```

Output

Column1
[1,"2","00:00:02"]
[2,"4","00:00:04"]
[3,"6","00:00:06"]

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_command_line()

Article • 01/10/2023

Parses a Unicode command-line string and returns a dynamic array of the command-line arguments.

Syntax

`parse_command_line(`*command_line*, *parser_type*`)`

Parameters

Name	Type	Required	Description
command_line	string	✓	The command line value to parse.
parser_type	string	✓	The only value that is currently supported is `"windows"`, which parses the command line the same way as CommandLineToArgvW.

Returns

A dynamic array of the command-line arguments.

Example

Run the query

Kusto

```
print parse_command_line("echo \"hello world!\"", "windows")
```

Output

Result
["echo","hello world!"]

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_csv()

Article • 01/10/2023

Splits a given string representing a single record of comma-separated values and returns a string array with these values.

Syntax

`parse_csv(`*csv_text*`)`

Parameters

Name	Type	Required	Description
csv_text	string	✓	A single record of comma-separated values.

> ⓘ Note
>
> - Embedded line feeds, commas, and quotes may be escaped using the double quotation mark ('"').
> - This function doesn't support multiple records per row (only the first record is taken).

Returns

A string array that contains the split values.

Examples

Filter by count of values in record

Count ADX conference sessions with more than three participants.

Run the query

```
Kusto
```

```
ConferenceSessions
| where array_length(parse_csv(participants)) > 3
| distinct *
```

Output

sessionid	...	participants
CON-PRT157	...	Guy Reginiano, Guy Yehudy, Pankaj Suri, Saeed Copty
BRK3099	...	Yoni Leibowitz, Eric Fleischman, Robert Pack, Avner Aharoni

Use escaping quotes

Run the query

Kusto

```
print result=parse_csv('aa,"b,b,b",cc,"Escaping quotes:
""Title""","line1\nline2"')
```

Output

result
["aa", "b,b,b", "cc", "Escaping quotes: "Title"", "line1\nline2"]

CSV with multiple records

Only the first record is taken since this function does not support multiple records.

Run the query

Kusto

```
print result_multi_record=parse_csv('record1,a,b,c\nrecord2,x,y,z')
```

Output

result_multi_record
["record1", "a", "b", "c"]

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_ipv4()

Article • 01/10/2023

Converts IPv4 string to a signed 64-bit wide long number representation in big-endian order.

Syntax

`parse_ipv4(`*ip*`)`

Parameters

Name	Type	Required	Description
ip	string	✓	The IPv4 that will be converted to long. The value may include net-mask using IP-prefix notation.

IP-prefix notation

IP-prefix notation (also known as CIDR notation) is a concise way of representing an IP address and its associated network mask. The format is `<base IP>/<prefix length>`, where the prefix length is the number of leading 1 bits in the netmask. The prefix length determines the range of IP addresses that belong to the network.

For IPv4, the prefix length is a number between 0 and 32. So the notation 192.168.2.0/24 represents the IP address 192.168.2.0 with a netmask of 255.255.255.0. This netmask has 24 leading 1 bits, or a prefix length of 24.

For IPv6, the prefix length is a number between 0 and 128. So the notation fe80::85d:e82c:9446:7994/120 represents the IP address fe80::85d:e82c:9446:7994 with a netmask of ffff:ffff:ffff:ffff:ffff:ffff:ffff:ff00. This netmask has 120 leading 1 bits, or a prefix length of 120.

Returns

If conversion is successful, the result will be a long number. If conversion isn't successful, the result will be `null`.

Example

Run the query

Kusto

```
datatable(ip_string: string)
[
    '192.168.1.1', '192.168.1.1/24', '255.255.255.255/31'
]
| extend ip_long = parse_ipv4(ip_string)
```

Output

ip_string	ip_long
192.168.1.1	3232235777
192.168.1.1/24	3232235776
255.255.255.255/31	4294967294

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_ipv4_mask()

Article • 01/10/2023

Converts the input string of IPv4 and netmask to a signed, 64-bit wide, long number representation in big-endian order.

Syntax

`parse_ipv4_mask(`*ip* `,` *prefix*`)`

Parameters

Name	Type	Required	Description
ip	string	✓	The IPv4 address to convert to a long number.
prefix	int	✓	An integer from 0 to 32 representing the number of most-significant bits that are taken into account.

Returns

If conversion is successful, the result will be a long number. If conversion isn't successful, the result will be `null`.

Example

Run the query

Kusto

```
print parse_ipv4_mask("127.0.0.1", 24)
```

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_ipv6()

Article • 01/10/2023

Converts IPv6 or IPv4 string to a canonical IPv6 string representation.

Syntax

`parse_ipv6(`*ip*`)`

Parameters

Name	Type	Required	Description
ip	string	✓	The IPv6/IPv4 network address that will be converted to canonical IPv6 representation. The value may include net-mask using IP-prefix notation.

IP-prefix notation

IP-prefix notation (also known as CIDR notation) is a concise way of representing an IP address and its associated network mask. The format is `<base IP>/<prefix length>`, where the prefix length is the number of leading 1 bits in the netmask. The prefix length determines the range of IP addresses that belong to the network.

For IPv4, the prefix length is a number between 0 and 32. So the notation 192.168.2.0/24 represents the IP address 192.168.2.0 with a netmask of 255.255.255.0. This netmask has 24 leading 1 bits, or a prefix length of 24.

For IPv6, the prefix length is a number between 0 and 128. So the notation fe80::85d:e82c:9446:7994/120 represents the IP address fe80::85d:e82c:9446:7994 with a netmask of ffff:ffff:ffff:ffff:ffff:ffff:ffff:ff00. This netmask has 120 leading 1 bits, or a prefix length of 120.

Returns

If conversion is successful, the result will be a string representing a canonical IPv6 network address. If conversion isn't successful, the result will be an empty string.

Example

Run the query

Kusto

```
datatable(ipv4: string)
[
    '192.168.255.255', '192.168.255.255/24', '255.255.255.255'
]
| extend ipv6 = parse_ipv6(ipv4)
```

Output

ipv4	ipv6
192.168.255.255	0000:0000:0000:0000:0000:ffff:c0a8:ffff
192.168.255.255/24	0000:0000:0000:0000:0000:ffff:c0a8:ff00
255.255.255.255	0000:0000:0000:0000:0000:ffff:ffff:ffff

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_ipv6_mask()

Article • 01/10/2023

Converts IPv6/IPv4 string and netmask to a canonical IPv6 string representation.

Syntax

`parse_ipv6_mask(`*ip*`,` *prefix*`)`

Parameters

Name	Type	Required	Description
ip	string		The IPv6/IPv4 network address to convert to canonical IPv6 representation. The value may include net-mask using IP-prefix notation.
prefix	int		An integer from 0 to 128 representing the number of most-significant bits that are taken into account.

IP-prefix notation

IP-prefix notation (also known as CIDR notation) is a concise way of representing an IP address and its associated network mask. The format is `<base IP>/<prefix length>`, where the prefix length is the number of leading 1 bits in the netmask. The prefix length determines the range of IP addresses that belong to the network.

For IPv4, the prefix length is a number between 0 and 32. So the notation 192.168.2.0/24 represents the IP address 192.168.2.0 with a netmask of 255.255.255.0. This netmask has 24 leading 1 bits, or a prefix length of 24.

For IPv6, the prefix length is a number between 0 and 128. So the notation fe80::85d:e82c:9446:7994/120 represents the IP address fe80::85d:e82c:9446:7994 with a netmask of ffff:ffff:ffff:ffff:ffff:ffff:ffff:ff00. This netmask has 120 leading 1 bits, or a prefix length of 120.

Returns

If conversion is successful, the result will be a string representing a canonical IPv6 network address. If conversion isn't successful, the result will be an empty string.

Example

Run the query

Kusto

```
datatable(ip_string: string, netmask: long)
[
    // IPv4 addresses
    '192.168.255.255', 120,  // 120-bit netmask is used
    '192.168.255.255/24', 124,  // 120-bit netmask is used, as IPv4 address
doesn't use upper 8 bits
    '255.255.255.255', 128,  // 128-bit netmask is used
    // IPv6 addresses
    'fe80::85d:e82c:9446:7994', 128,     // 128-bit netmask is used
    'fe80::85d:e82c:9446:7994/120', 124, // 120-bit netmask is used
    // IPv6 with IPv4 notation
    '::192.168.255.255', 128,  // 128-bit netmask is used
    '::192.168.255.255/24', 128,  // 120-bit netmask is used, as IPv4
address doesn't use upper 8 bits
]
| extend ip6_canonical = parse_ipv6_mask(ip_string, netmask)
```

Output

ip_string	netmask	ip6_canonical
192.168.255.255	120	0000:0000:0000:0000:0000:ffff:c0a8:ff00
192.168.255.255/24	124	0000:0000:0000:0000:0000:ffff:c0a8:ff00
255.255.255.255	128	0000:0000:0000:0000:0000:ffff:ffff:ffff
fe80::85d:e82c:9446:7994	128	fe80:0000:0000:0000:085d:e82c:9446:7994
fe80::85d:e82c:9446:7994/120	124	fe80:0000:0000:0000:085d:e82c:9446:7900
::192.168.255.255	128	0000:0000:0000:0000:0000:ffff:c0a8:ffff
::192.168.255.255/24	128	0000:0000:0000:0000:0000:ffff:c0a8:ff00

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_json()

Article • 03/12/2023

Interprets a `string` as a JSON value and returns the value as `dynamic`. If possible, the value is converted into relevant data types. For strict parsing with no data type conversion, use extract() or extract_json() functions.

It's better to use the parse_json() function over the extract_json() function when you need to extract more than one element of a JSON compound object. Use dynamic() when possible.

> **Deprecated aliases:** parsejson(), toobject(), todynamic()

Syntax

`parse_json(`*json*`)`

Parameters

Name	Type	Required	Description
json	string	✓	The string in the form of a JSON-formatted value or a dynamic property bag to parse as JSON.

Returns

An object of type `dynamic` that is determined by the value of *json*:

- If *json* is of type `dynamic`, its value is used as-is.
- If *json* is of type `string`, and is a properly formatted JSON string, then the string is parsed, and the value produced is returned.
- If *json* is of type `string`, but it isn't a properly formatted JSON string, then the returned value is an object of type `dynamic` that holds the original `string` value.

Example

In the following example, when `context_custom_metrics` is a `string` that looks like this:

```
JSON
```

```
{"duration":
{"value":118.0,"count":5.0,"min":100.0,"max":150.0,"stdDev":0.0,"sampledValu
e":118.0,"sum":118.0}}
```

then the following query retrieves the value of the `duration` slot in the object, and from that it retrieves two slots, `duration.value` and `duration.min` (`118.0` and `110.0`, respectively).

Kusto

```
T
| extend d=parse_json(context_custom_metrics)
| extend duration_value=d.duration.value, duration_min=d["duration"]["min"]
```

Notes

It's common to have a JSON string describing a property bag in which one of the "slots" is another JSON string.

For example:

Kusto

```
let d='{"a":123, "b":"{\\"c\\":456}"}';
print d
```

In such cases, it isn't only necessary to invoke `parse_json` twice, but also to make sure that in the second call, `tostring` is used. Otherwise, the second call to `parse_json` will just pass on the input to the output as-is, because its declared type is `dynamic`.

Kusto

```
let d='{"a":123, "b":"{\\"c\\":456}"}';
print d_b_c=parse_json(tostring(parse_json(d).b)).c
```

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_path()

Article • 01/26/2023

Parses a file path `string` and returns a dynamic object that contains the following parts of the path:

- Scheme
- RootPath
- DirectoryPath
- DirectoryName
- Filename
- Extension
- AlternateDataStreamName

In addition to the simple paths with both types of slashes, the function supports paths with:

- Schemas. For example, "file://..."
- Shared paths. For example, "\shareddrive\users..."
- Long paths. For example, "\?\C:..."
- Alternate data streams. For example, "file1.exe:file2.exe"

Syntax

`parse_path(`*path*`)`

Parameters

Name	Type	Required	Description
path	string	✓	The file path.

Returns

An object of type `dynamic` that included the path components as listed above.

Example

Run the query

Kusto

```
datatable(p:string)
[
    @"C:\temp\file.txt",
    @"temp\file.txt",
    "file://C:/temp/file.txt:some.exe",
    @"\\shared\users\temp\file.txt.gz",
    "/usr/lib/temp/file.txt"
]
| extend path_parts = parse_path(p)
```

Output

p	path_parts
C:\temp\file.txt	{"Scheme":"","RootPath":"C:","DirectoryPath":"C:\temp","DirectoryName":"temp","Filename":"file.txt","Extension":"txt","AlternateDataStr
temp\file.txt	{"Scheme":"","RootPath":"","DirectoryPath":"temp","DirectoryName":"temp","Filename":"file.txt","Extension":"txt","AlternateDataStream
file://C:/temp/file.txt:some.exe	{"Scheme":"file","RootPath":"C:","DirectoryPath":"C:/temp","DirectoryName":"temp","Filename":"file.txt","Extension":"txt","AlternateData
\shared\users\temp\file.txt.gz	{"Scheme":"","RootPath":"","DirectoryPath":"\\shared\users\temp","DirectoryName":"temp","Filename":"file.txt.gz","Extension":"gz","Alt
/usr/lib/temp/file.txt	{"Scheme":"","RootPath":"","DirectoryPath":"/usr/lib/temp","DirectoryName":"temp","Filename":"file.txt","Extension":"txt","AlternateData

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_url()

Article • 06/12/2023

Parses an absolute URL `string` and returns a `dynamic` object contains URL parts.

> **Deprecated aliases:** parseurl()

Syntax

`parse_url(`*url*`)`

Parameters

Name	Type	Required	Description
url	string	✓	An absolute URL, including its scheme, or the query part of the URL. For example, use the absolute `https://bing.com` instead of `bing.com`.

Returns

An object of type dynamic that included the URL components: Scheme, Host, Port, Path, Username, Password, Query Parameters, Fragment.

Example

Run the query

Kusto

```
print Result=parse_url("scheme://username:password@host:1234/this/is/a/path?k1=v1&k2=v2#fragment")
```

Output

Result
{"Scheme":"scheme", "Host":"host", "Port":"1234", "Path":"this/is/a/path", "Username":"username", "Password":"password", "Query Parameters":"{"k1":"v1", "k2":"v2"}", "Fragment":"fragment"}

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_urlquery()

Article • 01/26/2023

Returns a `dynamic` object that contains the query parameters.

> **Deprecated aliases:** parseurlquery()

Syntax

`parse_urlquery(`*query*`)`

Parameters

Name	Type	Required	Description
query	string	✓	The query part of the URL. The format must follow URL query standards (key=value& ...).

Returns

An object of type dynamic that includes the query parameters.

Examples

Run the query

Kusto

```
print Result=parse_urlquery("k1=v1&k2=v2&k3=v3")
```

Output

Result
{ "Query Parameters":"{"k1":"v1", "k2":"v2", "k3":"v3"}" }

The following example uses a function to extract specific query parameters.

Run the query

Kusto

```
let getQueryParamValue = (querystring: string, param: string) {
    let params = parse_urlquery(querystring);
    tostring(params["Query Parameters"].[param])
};
print UrlQuery = 'view=vs-2019&preserve-view=true'
| extend view = getQueryParamValue(UrlQuery, 'view')
| extend preserve = getQueryParamValue(UrlQuery, 'preserve-view')
```

Output

UrlQuery	view	preserve
view=vs-2019&preserve-view=true	vs-2019	true

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_user_agent()

Article • 01/26/2023

Interprets a user-agent string, which identifies the user's browser and provides certain system details to servers hosting the websites the user visits. The result is returned as dynamic.

Syntax

`parse_user_agent(`*user-agent-string*, *look-for*`)`

Parameters

Name	Type	Required	Description
user-agent-string	string	✓	The user-agent string to parse.
look-for	string or dynamic	✓	The value to search for in *user-agent-string*. The possible options are "browser", "os", or "device". If only a single parsing target is required, it can be passed a `string` parameter. If two or three targets are required, they can be passed as a `dynamic` array.

Returns

An object of type `dynamic` that contains the information about the requested parsing targets.

Browser: Family, MajorVersion, MinorVersion, Patch

OperatingSystem: Family, MajorVersion, MinorVersion, Patch, PatchMinor

Device: Family, Brand, Model

> ⚠ Warning
>
> The function implementation is built on regex checks of the input string against a huge number of predefined patterns. Therefore the expected time and CPU consumption is high. When the function is used in a query, make sure it runs in a distributed manner on multiple machines. If queries with this function are

frequently used, you may want to pre-create the results via **update policy**, but you need to take into account that using this function inside the update policy will increase the ingestion latency.

Examples

Look-for parameter as string

Run the query

Kusto

```
print useragent = "Mozilla/5.0 (Windows; U; en-US) AppleWebKit/531.9 (KHTML,
like Gecko) AdobeAIR/2.5.1"
| extend x = parse_user_agent(useragent, "browser")
```

Expected result is a dynamic object:

{ "Browser": { "Family": "AdobeAIR", "MajorVersion": "2", "MinorVersion": "5", "Patch": "1" } }

Look-for parameter as dynamic array

Run the query

Kusto

```
print useragent = "Mozilla/5.0 (SymbianOS/9.2; U; Series60/3.1 NokiaN81-
3/10.0.032 Profile/MIDP-2.0 Configuration/CLDC-1.1 ) AppleWebKit/413 (KHTML,
like Gecko) Safari/4"
| extend x = parse_user_agent(useragent, dynamic(["browser","os","device"]))
```

Expected result is a dynamic object:

{ "Browser": { "Family": "Nokia OSS Browser", "MajorVersion": "3", "MinorVersion": "1", "Patch": "" }, "OperatingSystem": { "Family": "Symbian OS", "MajorVersion": "9", "MinorVersion": "2", "Patch": "", "PatchMinor": "" }, "Device": { "Family": "Nokia N81", "Brand": "Nokia", "Model": "N81-3" } }

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_version()

Article • 01/26/2023

Converts the input string representation of the version to a comparable decimal number.

Syntax

`parse_version` (*version*)

Parameters

Name	Type	Required	Description
version	string	✓	The version to be parsed.

> ⓘ Note
>
> - *version* must contain from one to four version parts, represented as numbers and separated with dots ('.').
> - Each part of *version* may contain up to eight digits, with the max value at 99999999.
> - If the number of parts is less than four, all the missing parts are considered as trailing. For example, `1.0 == 1.0.0.0`.

Returns

If conversion is successful, the result will be a decimal. If conversion is unsuccessful, the result will be `null`.

Example

Run the query

```
Kusto
```

```
let dt = datatable(v: string)
    [
    "0.0.0.5", "0.0.7.0", "0.0.3", "0.2", "0.1.2.0", "1.2.3.4", "1",
"99999999.0.0.0"
];
dt
| project v1=v, _key=1
| join kind=inner (dt | project v2=v, _key = 1) on _key
| where v1 != v2
| summarize v1 = max(v1), v2 = min(v2) by (hash(v1) + hash(v2)) // removing
duplications
| project v1, v2, higher_version = iif(parse_version(v1) >
parse_version(v2), v1, v2)
```

Output

v1	v2	higher_version
99999999.0.0.0	0.0.0.5	99999999.0.0.0
1	0.0.0.5	1
1.2.3.4	0.0.0.5	1.2.3.4
0.1.2.0	0.0.0.5	0.1.2.0
0.2	0.0.0.5	0.2
0.0.3	0.0.0.5	0.0.3
0.0.7.0	0.0.0.5	0.0.7.0
99999999.0.0.0	0.0.7.0	99999999.0.0.0
1	0.0.7.0	1
1.2.3.4	0.0.7.0	1.2.3.4
0.1.2.0	0.0.7.0	0.1.2.0
0.2	0.0.7.0	0.2
0.0.7.0	0.0.3	0.0.7.0
99999999.0.0.0	0.0.3	99999999.0.0.0
1	0.0.3	1
1.2.3.4	0.0.3	1.2.3.4
0.1.2.0	0.0.3	0.1.2.0

v1	v2	higher_version
0.2	0.0.3	0.2
99999999.0.0.0	0.2	99999999.0.0.0
1	0.2	1
1.2.3.4	0.2	1.2.3.4
0.2	0.1.2.0	0.2
99999999.0.0.0	0.1.2.0	99999999.0.0.0
1	0.1.2.0	1
1.2.3.4	0.1.2.0	1.2.3.4
99999999.0.0.0	1.2.3.4	99999999.0.0.0
1.2.3.4	1	1.2.3.4
99999999.0.0.0	1	99999999.0.0.0

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

parse_xml()

Article • 01/26/2023

Interprets a `string` as an XML value, converts the value to a JSON, and returns the value as `dynamic`.

Syntax

`parse_xml(`*xml*`)`

Parameters

Name	Type	Required	Description
xml	string	✓	The XML-formatted string value to parse.

Returns

An object of type dynamic that is determined by the value of *xml*, or null, if the XML format is invalid.

The conversion is done as follows:

XML	JSON	Access
`<e/>`	{ "e": null }	o.e
`<e>text</e>`	{ "e": "text" }	o.e
`<e name="value" />`	{ "e":{"@name": "value"} }	o.e["@name"]
`<e name="value">text</e>`	{ "e": { "@name": "value", "#text": "text" } }	o.e["@name"] o.e["#text"]
`<e> <a>text</a> <b>text</b> </e>`	{ "e": { "a": "text", "b": "text" } }	o.e.a o.e.b
`<e> <a>text</a> <a>text</a> </e>`	{ "e": { "a": ["text", "text"] } }	o.e.a[0] o.e.a[1]
`<e> text <a>text</a> </e>`	{ "e": { "#text": "text", "a": "text" } }	1`o.e["#text"] o.e.a

> ⓘ Note
>
> - Maximal input `string` length for `parse_xml` is 1 MB (1,048,576 bytes). Longer strings interpretation will result in a null object.
> - Only element nodes, attributes and text nodes will be translated. Everything else will be skipped.

Example

In the following example, when `context_custom_metrics` is a `string` that looks like this:

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<duration>
    <value>118.0</value>
    <count>5.0</count>
    <min>100.0</min>
    <max>150.0</max>
    <stdDev>0.0</stdDev>
    <sampledValue>118.0</sampledValue>
    <sum>118.0</sum>
</duration>
```

then the following CSL Fragment translates the XML to the following JSON:

JSON

```
{
    "duration": {
        "value": 118.0,
        "count": 5.0,
        "min": 100.0,
        "max": 150.0,
        "stdDev": 0.0,
        "sampledValue": 118.0,
        "sum": 118.0
    }
}
```

and retrieves the value of the `duration` slot in the object, and from that it retrieves two slots, `duration.value` and `duration.min` (`118.0` and `100.0`, respectively).

Kusto

```
T
| extend d=parse_xml(context_custom_metrics)
| extend duration_value=d.duration.value, duration_min=d["duration"]["min"]
```

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

percentile_tdigest()

Article • 03/12/2023

Calculates the percentile result from the `tdigest` results (which was generated by tdigest() or tdigest_merge())

Syntax

`percentile_tdigest(`*expr*`,` *percentile1* `,` *typeLiteral*`)`

Parameters

Name	Type	Required	Description
expr	string	✓	An expression that was generated by tdigest or tdigest_merge().
percentile	long	✓	The value that specifies the percentile.
typeLiteral	string		A type literal. If provided, the result set will be of this type. For example, `typeof(long)` will cast all results to type `long`.

Returns

The percentile value of each value in *expr*.

> **Tip**
>
> - If the type was provided, the result will be a column of the same type provided with the results of the percentile. In this case, all `tdigest` functions must be of that type.
> - If *expr* includes `tdigest` functions of different types, don't provide the type. The result will be of type dynamic. See below examples.

Examples

Run the query

Kusto

```
StormEvents
| summarize tdigestRes = tdigest(DamageProperty) by State
| project percentile_tdigest(tdigestRes, 100)
```

Output

percentile_tdigest_tdigestRes
0
62000000
110000000
1200000
250000

Run the query

Kusto

```
StormEvents
| summarize tdigestRes = tdigest(DamageProperty) by State
| union (StormEvents | summarize tdigestRes = tdigest(EndTime) by State)
| project percentile_tdigest(tdigestRes, 100)
```

Output

percentile_tdigest_tdigestRes
[0]
[62000000]
["2007-12-20T11:30:00.0000000Z"]
["2007-12-31T23:59:00.0000000Z"]

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

percentile_array_tdigest()

Article • 01/26/2023

Calculates the percentile result from the `tdigest` results (which was generated by tdigest() or tdigest_merge())

Syntax

`percentiles_array_tdigest(`*tdigest*`,` *percentile1* `[,` *percentile2*`, ...])`

`percentiles_array_tdigest(`*tdigest*`,` *Dynamic array* `[,` *typeLiteral* `])`

Parameters

Name	Type	Required	Description
tdigest	string	✓	The tdigest or tdigest_merge() results used to calculate the percentiles.
percentile	real	✓	A value or comma-separated list of values that specifies the percentiles.
Dynamic array	dynamic	✓	A dynamic array of real numbers that specify the percentiles.
typeLiteral	string		A type literal. For example, `typeof(long)`. If provided, the result set will be of this type.

Returns

The percentile/percentiles value of each value in *tdigest*.

> **Tip**
>
> - The function must receive at least one percent (and maybe more, see the syntax above: *percentile1* `[,` *percentile2*] ...[`,` *percentileN*]) and the result will be a dynamic array that includes the results. (such like **percentiles**())
> - If only one percent was provided, and the type was provided also, then the result will be a column of the same type provided with the results of that percent. In this case, all `tdigest` functions must be of that type.

- If *tdigest* includes `tdigest` functions of different types, don't provide the type. The result will be of type dynamic. See below examples.

Examples

Run the query

Kusto

```
StormEvents
| summarize tdigestRes = tdigest(DamageProperty) by State
| project percentiles_array_tdigest(tdigestRes, range(0, 100, 50),
typeof(int))
```

Output

percentile_tdigest_tdigestRes
[0,0,0]
[0,0,62000000]
[0,0,110000000]
[0,0,1200000]
[0,0,250000]

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

percentrank_tdigest()

Article • 03/12/2023

Calculates the approximate rank of the value in a set, where rank is expressed as a percentage of the set's size. This function can be viewed as the inverse of the percentile.

Syntax

`percentrank_tdigest(`*digest*`,` *value*`)`

Parameters

Name	Type	Required	Description
digest	string	✓	An expression that was generated by tdigest() or tdigest_merge().
value	scalar	✓	An expression representing a value to be used for percentage ranking calculation.

> ⓘ Note
>
> The type of *value* and the type of the elements in *digest* should be the same.

Returns

The percentage rank of value in a dataset.

Examples

Getting the percentrank_tdigest() of the damage property that valued 4490$ is ~85%:

Run the query

Kusto

```
StormEvents
| summarize tdigestRes = tdigest(DamageProperty)
| project percentrank_tdigest(tdigestRes, 4490)
```

Output

Column1
85.0015237192293

Using percentile 85 over the damage property should give 4490$:

Run the query

Kusto

```
StormEvents
| summarize tdigestRes = tdigest(DamageProperty)
| project percentile_tdigest(tdigestRes, 85, typeof(long))
```

Output

percentile_tdigest_tdigestRes
4490

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

pi()

Article • 01/16/2023

Returns the constant value of Pi.

Syntax

```
pi()
```

Returns

The double value of Pi (3.1415926...)

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

pow()

Article • 01/26/2023

Returns a result of raising to power

Syntax

`pow(` *base* `,` *exponent* `)`

Parameters

Name	Type	Required	Description
base	int, real, or long	✓	The base value.
exponent	int, real, or long	✓	The exponent value.

Returns

Returns base raised to the power exponent: base ^ exponent.

Example

Run the query

Kusto

```
print result=pow(2, 3)
```

Output

result
8

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

punycode_from_string()

Article • 05/30/2023

Encodes input string to Punycode form. The result string contains only ASCII characters. The result string doesn't start with "xn--".

Syntax

`punycode_from_string('input_string')`

Parameters

Name	Type	Required	Description
input_string	`string`	✓	A string to be encoded to punycode form. The function accepts one string argument.

Returns

- Returns a `string` that represents punycode-encoded original string.
- Returns an empty result if encoding failed.

Examples

Run the query

Kusto

```
print encoded = punycode_from_string('académie-française')
```

encoded
acadmie-franaise-npb1a

Run the query

Kusto

```
print domain='艺术.com'
| extend domain_vec = split(domain, '.')
| extend encoded_host = punycode_from_string(tostring(domain_vec[0]))
| extend encoded_domain = strcat('xn--', encoded_host, '.', domain_vec[1])
```

domain	domain_vec	encoded_host	encoded_domain
艺术.com	["艺术","com"]	cqv902d	xn--cqv902d.com

Next steps

Use punycode_to_string() to retrieve the original decoded string.

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

punycode_to_string()

Article • 05/31/2023

Decodes input string from punycode form. The string shouldn't contain the initial xn--, and must contain only ASCII characters.

Syntax

```
punycode_to_string('input_string')
```

Parameters

Name	Type	Required	Description
input_string	`string`	✓	A string to be decoded from punycode form. The function accepts one string argument.

Returns

- Returns a `string` that represents the original, decoded string.
- Returns an empty result if decoding failed.

Example

Run the query

Kusto

```
print decoded = punycode_to_string('acadmie-franaise-npb1a')
```

decoded
académie-française

Next steps

Use punycode_from_string() to encode a string to punycode form.

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

radians()

Article • 01/31/2023

Converts angle value in degrees into value in radians, using formula `radians = (PI / 180 ) * angle_in_degrees`

Syntax

`radians(`*degrees*`)`

Parameters

Name	Type	Required	Description
degrees	real	✓	The angle in degrees.

Returns

The corresponding angle in radians for an angle specified in degrees.

Example

Run the query

Kusto

```
print radians0 = radians(90), radians1 = radians(180), radians2 =
radians(360)
```

Output

radians0	radians1	radians2
1.5707963267949	3.14159265358979	6.28318530717959

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

rand()

Article • 01/16/2023

Returns a random number.

```Kusto
rand()
rand(1000)
```

Syntax

- `rand()` - returns a value of type `real` with a uniform distribution in the range [0.0, 1.0).
- `rand(` *N* `)` - returns a value of type `real` chosen with a uniform distribution from the set {0.0, 1.0, ..., *N* - 1}.

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

range()

Article • 03/12/2023

Generates a dynamic array holding a series of equally spaced values.

Syntax

`range(`*start*`,` *stop* `[,` *step*`])`

Parameters

Name	Type	Required	Description
start	scalar	✓	The value of the first element in the resulting array.
stop	scalar	✓	The value of the last element in the resulting array, or the least value that is greater than the last element in the resulting array and within an integer multiple of *step* from *start*.
step	scalar		The difference between two consecutive elements of the array. The default value for *step* is `1` for numeric and `1h` for `timespan` or `datetime`.

Returns

Dynamic array whose values are: *start*, *start* + *step*, ... up to and including *stop*. The array will be truncated if the maximum number of values is reached.

> ⓘ **Note**
>
> The maximum number of values is 1,048,576 (2^20).

Examples

The following example returns `[1, 4, 7]`:

Kusto

```
T | extend r = range(1, 8, 3)
```

The following example returns an array holding all days in the year 2015:

```Kusto
T | extend r = range(datetime(2015-01-01), datetime(2015-12-31), 1d)
```

The following example returns `[1,2,3]`:

```Kusto
range(1, 3)
```

The following example returns `["01:00:00","02:00:00","03:00:00","04:00:00","05:00:00"]`:

```Kusto
range(1h, 5h)
```

The following example returns `1048576`:

```Kusto
print r = range(1,1000000000) | mv-expand r | count
```

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

rank_tdigest()

Article • 03/12/2023

Calculates the approximate rank of the value in a set. Rank of value `v` in a set `S` is defined as count of members of `S` that are smaller or equal to `v`, `S` is represented by its `tdigest`.

Syntax

`rank_tdigest(`*digest*`,` *value*`)`

Parameters

Name	Type	Required	Description
digest	string	An expression that was generated by tdigest() or tdigest_merge().	
value	scalar	An expression representing a value to be used for ranking calculation.	

Returns

The rank foreach value in a data set.

> **Tip**
>
> The values that you want to get its rank must be of the same type as the `tdigest`.

Examples

In a sorted list (1-1000), the rank of 685 is its index:

Run the query

Kusto

```
range x from 1 to 1000 step 1
| summarize t_x=tdigest(x)
| project rank_of_685=rank_tdigest(t_x, 685)
```

Output

rank_of_685
685

This query calculates the rank of value 4490$ over all damage properties costs:

Run the query

Kusto

```
StormEvents
| summarize tdigestRes = tdigest(DamageProperty)
| project rank_of_4490=rank_tdigest(tdigestRes, 4490)
```

Output

rank_of_4490
50207

Getting the estimated percentage of the rank (by dividing by the set size):

Run the query

Kusto

```
StormEvents
| summarize tdigestRes = tdigest(DamageProperty), count()
| project rank_tdigest(tdigestRes, 4490) * 100.0 / count_
```

Output

Column1
85.0015237192293

The percentile 85 of the damage properties costs is 4490$:

Run the query

Kusto

```
StormEvents
| summarize tdigestRes = tdigest(DamageProperty)
| project percentile_tdigest(tdigestRes, 85, typeof(long))
```

Output

percentile_tdigest_tdigestRes
4490

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

regex_quote()

Article • 01/31/2023

Returns a string that escapes all regular expression characters.

Syntax

`regex_quote(`*string*`)`

Parameters

Name	Type	Required	Description
string	string	✓	The string to escape.

Returns

Returns *string* where all regex expression characters are escaped.

Example

Run the query

Kusto

```
print result = regex_quote('(so$me.Te^xt)')
```

Output

result
`\(so\\$me\\.Te\\^xt\\)`

Feedback

Was this page helpful? Yes No

repeat()

Article • 01/31/2023

Generates a dynamic array containing a series comprised of repeated numbers.

Syntax

`repeat(`*value*`,` *count*`)`

Parameters

Name	Type	Required	Description
value	bool, int, long, real, datetime, string or timespan	✓	The value of the element in the resulting array.
count	int	✓	The count of the elements in the resulting array.

Returns

If *count* is equal to zero, an empty array is returned. If *count* is less than zero, a null value is returned.

Examples

The following example returns `[1, 1, 1]`:

Kusto

```
T | extend r = repeat(1, 3)
```

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

replace_regex()

Article • 06/07/2023

Replaces all regex matches with a specified pattern.

> **Deprecated aliases:** replace()

Syntax

`replace_regex(`*source*`,`*lookup_regex*`,` *rewrite_pattern*`)`

Parameters

Name	Type	Required	Description
source	string	✓	The text to search and replace.
lookup_regex	string	✓	The regular expression to search for in *text*. The expression can contain capture groups in parentheses.
rewrite_pattern	string	✓	The replacement regex for any match made by *matchingRegex*. Use `\0` to refer to the whole match, `\1` for the first capture group, `\2` and so on for subsequent capture groups.

Returns

Returns the *source* after replacing all matches of *lookup_regex* with evaluations of *rewrite_pattern*. Matches do not overlap.

Example

Run the query

Kusto

```
range x from 1 to 5 step 1
| extend str=strcat('Number is ', tostring(x))
| extend replaced=replace_regex(str, @'is (\d+)', @'was: \1')
```

Output

x	str	replaced
1	Number is 1.000000	Number was: 1.000000
2	Number is 2.000000	Number was: 2.000000
3	Number is 3.000000	Number was: 3.000000
4	Number is 4.000000	Number was: 4.000000
5	Number is 5.000000	Number was: 5.000000

See also

- To replace a single string, see replace_string().
- To replace multiple strings, see replace_strings().
- To replace a set of characters, see translate().

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

replace_string()

Article • 06/07/2023

Replaces all string matches with a specified string.

> **Deprecated aliases:** replace()

To replace multiple strings, see replace_strings().

Syntax

`replace_string(`*text*`,` *lookup*`,` *rewrite*`)`

Parameters

Name	Type	Required	Description
text	string	✓	The source string.
lookup	string	✓	The string to be replaced.
rewrite	string	✓	The replacement string.

Returns

Returns the *text* after replacing all matches of *lookup* with evaluations of *rewrite*. Matches don't overlap.

Example

Run the query

Kusto

```
range x from 1 to 5 step 1
| extend str=strcat('Number is ', tostring(x))
| extend replaced=replace_string(str, 'is', 'was')
```

Output:

x	str	replaced
1	Number is 1.000000	Number was 1.000000
2	Number is 2.000000	Number was 2.000000
3	Number is 3.000000	Number was 3.000000
4	Number is 4.000000	Number was 4.000000
5	Number is 5.000000	Number was 5.000000

See also

- To replace multiple strings, see replace_strings().
- To replace strings based on regular expression, see replace_regex().
- To replace a set of characters, see translate().

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

replace_strings()

Article • 06/07/2023

Replaces all strings matches with specified strings.

To replace an individual string, see replace_string().

Syntax

`replace_strings(`*text*`,` *lookups*`,` *rewrites*`)`

Parameters

Name	Type	Required	Description
text	string	✓	The source string.
lookups	dynamic	✓	The array that includes lookup strings. Array element that isn't a string is ignored.
rewrites	dynamic	✓	The array that includes rewrites. Array element that isn't a string is ignored (no replacement made).

Returns

Returns *text* after replacing all matches of *lookups* with evaluations of *rewrites*. Matches don't overlap.

Examples

Simple replacement

Run the query

Kusto

```
print Message="A magic trick can turn a cat into a dog"
| extend Outcome = replace_strings(
        Message,
        dynamic(['cat', 'dog']), // Lookup strings
```

```
        dynamic(['dog', 'pigeon']) // Replacements
        )
```

Message	Outcome
A magic trick can turn a cat into a dog	A magic trick can turn a dog into a pigeon

Replacement with an empty string

Replacement with an empty string removes the matching string.

Run the query

Kusto

```
print Message="A magic trick can turn a cat into a dog"
| extend Outcome = replace_strings(
        Message,
        dynamic(['turn', ' into a dog']), // Lookup strings
        dynamic(['disappear', '']) // Replacements
        )
```

Message	Outcome
A magic trick can turn a cat into a dog	A magic trick can disappear a cat

Replacement order

The order of match elements matters: the earlier match takes the precedence. Note the difference between Outcome1 and Outcome2: `This` vs `Thwas`.

Run the query

Kusto

```
 print Message="This is an example of using replace_strings()"
| extend Outcome1 = replace_strings(
        Message,
        dynamic(['This', 'is']), // Lookup strings
        dynamic(['This', 'was']) // Replacements
        ),
        Outcome2 = replace_strings(
        Message,
        dynamic(['is', 'This']), // Lookup strings
```

```
        dynamic(['was', 'This']) // Replacements
        )
```

Message	Outcome1	Outcome2
This is an example of using replace_strings()	This was an example of using replace_strings()	Thwas was an example of using replace_strings()

Nonstring replacement

Replace elements that aren't strings aren't replaced and the original string is kept. The match is still considered being valid, and other possible replacements aren't performed on the matched string. In the following example, 'This' isn't replaced with the numeric `12345`, and it remains in the output unaffected by possible match with 'is'.

Run the query

Kusto

```
 print Message="This is an example of using replace_strings()"
| extend Outcome = replace_strings(
        Message,
        dynamic(['This', 'is']), // Lookup strings
        dynamic([12345, 'was']) // Replacements
        )
```

Message	Outcome
This is an example of using replace_strings()	This was an example of using replace_strings()

See also

- For a replacement of a single string, see replace_string().
- For a replacement based on regular expression, see replace_regex().
- For replacing a set of characters, see translate().

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

reverse()

Article • 01/31/2023

Function reverses the order of the input string. If the input value isn't of type `string`, then the function forcibly casts the value to type `string`.

Syntax

`reverse(`*value*`)`

Parameters

Name	Type	Required	Description
value	string	✓	input value.

Returns

The reverse order of a string value.

Examples

Run the query

Kusto

```
print str = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
| extend rstr = reverse(str)
```

Output

str	rstr
ABCDEFGHIJKLMNOPQRSTUVWXYZ	ZYXWVUTSRQPONMLKJIHGFEDCBA

Run the query

Kusto

```
print ['int'] = 12345, ['double'] = 123.45,
['datetime'] = datetime(2017-10-15 12:00), ['timespan'] = 3h
| project rint = reverse(['int']), rdouble = reverse(['double']),
rdatetime = reverse(['datetime']), rtimespan = reverse(['timespan'])
```

Output

rint	rdouble	rdatetime	rtimespan
54321	54.321	Z0000000.00:00:21T51-01-7102	00:00:30

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

round()

Article • 02/16/2023

Returns the rounded number to the specified precision.

Syntax

`round(`*number* [`,` *precision*]`)`

Parameters

Name	Type	Required	Description
number	long or real	✓	The number to calculate the round on.
precision	int		The number of digits to round to. The default is 0.

Returns

The rounded number to the specified precision.

Round is different from the bin() function in that the `round()` function rounds a number to a specific number of digits while the `bin()` function rounds the value to an integer multiple of a given bin size. For example, `round(2.15, 1)` returns 2.2 while `bin(2.15, 1)` returns 2.

Examples

Kusto

```
round(2.98765, 3)   // 2.988
round(2.15, 1)      // 2.2
round(2.15)         // 2 // equivalent to round(2.15, 0)
round(-50.55, -2)   // -100
round(21.5, -1)     // 20
```

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

set_difference()

Article • 02/06/2023

Returns a `dynamic` (JSON) array of the set of all distinct values that are in the first array but aren't in other arrays - (((arr1 \ arr2) \ arr3) \ ...).

Syntax

`set_difference(`*set1*`,` *set2* `[,`*set3*, ...`])`

Parameters

Name	Type	Required	Description
set1...*setN*	dynamic	✓	Arrays used to create a difference set. A minimum of 2 arrays are required. See pack_array.

Returns

Returns a dynamic array of the set of all distinct values that are in *set1* but aren't in other arrays.

Example

Run the query

Kusto

```
range x from 1 to 3 step 1
| extend y = x * 2
| extend z = y * 2
| extend w = z * 2
| extend a1 = pack_array(x,y,x,z), a2 = pack_array(x, y), a3 =
pack_array(x,y,w)
| project set_difference(a1, a2, a3)
```

Output

Column1

Column1
[4]
[8]
[12]

Run the query

```
Kusto
print arr = set_difference(dynamic([1,2,3]), dynamic([1,2,3]))
```

Output

arr
[]

See also

- set_union()
- set_intersect()

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

set_has_element()

Article • 02/06/2023

Determines whether the specified set contains the specified element.

Syntax

`set_has_element(`*set*`,` *value*`)`

Parameters

Name	Type	Required	Description
set	dynamic	✓	The input array to search.
value		✓	The value for which to search. The value should be of type `long`, `int`, `double`, `datetime`, `timespan`, `decimal`, `string`, `guid`, or `bool`.

Returns

`true` or `false` depending on if the value exists in the array.

Example

Run the query

Kusto

```
print arr=dynamic(["this", "is", "an", "example"])
| project Result=set_has_element(arr, "example")
```

Output

Result
true

See also

Use array_index_of(arr, value) to find the position at which the value exists in the array. Both functions are equally performant.

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

set_intersect()

Article • 02/06/2023

Returns a `dynamic` array of the set of all distinct values that are in all arrays - (arr1 ∩ arr2 ∩ ...).

Syntax

`set_intersect(`*set1*`,` *set2* `[,` *set3*, ...`])`

Parameters

Name	Type	Required	Description
set1...setN	dynamic	✓	Arrays used to create an intersect set. A minimum of 2 arrays are required. See pack_array.

Returns

Returns a dynamic array of the set of all distinct values that are in all arrays.

Example

Run the query

Kusto

```
range x from 1 to 3 step 1
| extend y = x * 2
| extend z = y * 2
| extend w = z * 2
| extend a1 = pack_array(x,y,x,z), a2 = pack_array(x, y), a3 =
pack_array(w,x)
| project set_intersect(a1, a2, a3)
```

Output

Column1
[1]

Column1
[2]
[3]

Run the query

```
Kusto
print arr = set_intersect(dynamic([1, 2, 3]), dynamic([4,5]))
```

Output

arr
[]

See also

- set_union()
- set_difference()

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

set_union()

Article • 04/04/2023

Returns a `dynamic` array of the set of all distinct values that are in any of the arrays - (arr1 ∪ arr2 ∪ ...).

Syntax

`set_union(`*set1*`,` *set2* `[,` *set3*, ...`])`

Parameters

Name	Type	Required	Description
set1...setN	dynamic	✓	Arrays used to create a union set. A minimum of 2 arrays are required. See pack_array.

Returns

Returns a dynamic array of the set of all distinct values that are in any of arrays.

Example

Set from multiple dynamic array

Run the query

Kusto

```
range x from 1 to 3 step 1
| extend y = x * 2
| extend z = y * 2
| extend w = z * 2
| extend a1 = pack_array(x,y,x,z), a2 = pack_array(x, y), a3 = pack_array(w)
| project a1,a2,a3,Out=set_union(a1, a2, a3)
```

Output

a1	a2	a3	`Out`

a1	a2	a3	Out
[1,2,1,4]	[1,2]	[8]	[1,2,4,8]
[2,4,2,8]	[2,4]	[16]	[2,4,8,16]
[3,6,3,12]	[3,6]	[24]	[3,6,12,24]

Set from one dynamic array

Run the query

Kusto

```
datatable (Arr1: dynamic)
[
    dynamic(['A4', 'A2', 'A7', 'A2']),
    dynamic(['C4', 'C7', 'C1', 'C4'])
]
| extend Out=set_union(Arr1, Arr1)
```

Output

Arr1	Out
["A4","A2","A7","A2"]	["A4","A2","A7"]
["C4","C7","C1","C4"]	["C4","C7","C1"]

See also

- set_intersect()
- set_difference()

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

sign()

Article • 02/06/2023

Returns the sign of the numeric expression.

Syntax

`sign(`*number*`)`

Parameters

Name	Type	Required	Description
number	real	✓	The number for which to return the sign.

Returns

The positive (+1), zero (0), or negative (-1) sign of the specified expression.

Examples

Run the query

Kusto

```
print s1 = sign(-42), s2 = sign(0), s3 = sign(11.2)
```

Output

s1	s2	s3
-1	0	1

Feedback

Was this page helpful? Yes No

sin()

Article • 03/06/2023

Returns the sine function value of the specified angle. The angle is specified in radians.

Syntax

`sin(`*number*`)`

Parameters

Name	Type	Required	Description
number	real	✓	The value in radians for which to calculate the sine.

Returns

The sine of *number* of radians.

Example

Kusto

```
print sin(1)
```

Output

result
0.841470984807897

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

split()

Article • 02/06/2023

The `split()` function takes a string and splits it into substrings based on a specified delimiter, returning the substrings in an array. Optionally, you can retrieve a specific substring by specifying its index.

Syntax

`split(`*source*`,` *delimiter* [`,` *requestedIndex*]`)`

Parameters

Name	Type	Required	Description
source	string	✓	The source string that will be split according to the given delimiter.
delimiter	string	✓	The delimiter that will be used in order to split the source string.
requestedIndex	int		A zero-based index. If provided, the returned string array will contain the requested substring at the index if it exists.

Returns

An array of substrings obtained by separating the *source* string by the specified *delimiter*, or a single substring at the specified *requestedIndex*.

Examples

Run the query

Kusto

```
print
    split("aa_bb", "_"),           // ["aa","bb"]
    split("aaa_bbb_ccc", "_", 1),  // ["bbb"]
    split("", "_"),                // [""]
    split("a__b", "_"),            // ["a","","b"]
    split("aabbcc", "bb")          // ["aa","cc"]
```

print_0	print_1	print_2	print_3	print4
["aa","bb"]	["bbb"]	[""]	["a","","b"]	["aa","cc"]

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A

sqrt()

Article • 02/06/2023

Returns the square root of the input.

Syntax

`sqrt(`*number*`)`

Parameters

Name	Type	Required	Description
number	int, long, or real	✓	The number for which to calculate the square root.

Returns

- A positive number such that `sqrt(x) * sqrt(x) == x`
- `null` if the argument is negative or can't be converted to a `real` value.

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A