

Structure keywords

from Point to a table for the query. Specify a table name using tags.

`from my.app.web.auth`

where Where clause to filter results.

`where [filter1 expression], [filter2 expression]`

`String` values in expressions have to be surrounded by double quotes, single quotes are not allowed.

select Add column to result set:

`select source_column or column operation as destination_column`

`select uriHost(uri) as host`

group every / by Group clause with an optional server and client aggregation period filter:

`group [every server_period] by column]`

`[every client_period]`

`group by statusCode`

`group every 10m`

`group every 10m by statusCode every 1m`

every Client aggregation filter:

`every period`

`Period` syntax: an integer number follow by a symbol indicating the time period: `[s:seconds, m:minutes, h:hours, d:days, no suffix :milliseconds - seconds]`

`every 0` means no client period.

`every 5m by serverIp`

ifthenelse `if/then/else` equivalent clause to set conditionally column values:

`ifthenelse(condition, error_value, success_value)`

`select ifthenelse(statusCode != 200, "Error", "Success") as statusCodeDesc`

Structure keywords (cont)

decode `switch/case` equivalent clause to set conditionally column values:

`decode(column, check_value, value, [check_value2, value2])`

Each pair of arguments `[check_value, value]` is equivalent to a case sentence of a switch statement.

`decode(statusCode, 200, "Success", 400, "Not Found", 406, "Error", 404, "Error") as statusCodeDesc`

nvl **Null-Coalescing** operator. Allow to set an alternate value when input value is null.

`nvl(column, alternate_value_when_null)`

Aggregation functions and operators

avg Returns the **average** of a range of values `[avg]` or only over *not null* values `[nnavg]` of the results on each group:

`avg(column); nnavg(column)`

count Returns the **count** of results on each group:

`count([column])`

With argument, include only not null entries in the count.

first / nnfirst Returns the **first** or the **not null first** entry of the results on each group:

`first(column); nnfirst(column)`



By **Rafa Hernández (elpluto)**
cheatography.com/elpluto/

Published 2nd January, 2023.

Last updated 2nd January, 2023.

Page 1 of 8.

Sponsored by **ApolloPad.com**

Everyone has a novel in them. Finish Yours!

<https://apollopod.com>

Aggregation functions and operators (cont)

last / nlast	Returns the last or the not null last entry of the results on each group: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>last(column); nlast(column)</code>
max / min	Returns the maximum or the minimum value for the columns provided, on each group: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>max/min(col1, [col2], [col3]...)</code>
median	Returns the statistical median for a column on each group: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>median(column)</code> <input type="checkbox"/> <input checked="" type="radio"/> Restricted to columns of integer type
sum	Returns the sum of the results on each group: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>sum(column)</code>
sum2	Returns the sum of the squares of the results on each group: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>sum2(column)</code>
percentile5 percentile10 percentile25 percentile75 percentile90 percentile95	Returns the specific statistic percentileN , using linear interpolation, of the results on each group: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>percentile[N](column)</code>
stddev / nnstddev	Returns the biased standard deviation [stddev] of the values or not null values [nnstddev] of the results on each group: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>stddev / nnstddev(column)</code> <input type="checkbox"/> <input checked="" type="radio"/> Biased

Aggregation functions and operators (cont)

ustddev / nnstddev	Returns the unbiased standard deviation [ustddev] of the values or not null values [nnstddev] of the results on each group: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>ustddev / nnstddev(column)</code> <input type="checkbox"/> <input checked="" type="radio"/> Unbiased
var / nnvar	Returns the biased variance [var] of the values or not null values [nnvar] of the results on each group: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>var/nnvar(column)</code> <input type="checkbox"/> <input checked="" type="radio"/> Biased
uvar / nnuvar	Returns the unbiased variance [uvar] of the values or not null values [nnuvar] of the results on each group: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>uvar/nnuvar(column)</code> <input type="checkbox"/> <input checked="" type="radio"/> Unbiased
hllpp	Returns the estimated count of distinct values of the results on each group using the HyperLogLog++ algorithm: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>hllpp(column)</code> <input type="checkbox"/> <input checked="" type="radio"/> Applies on DC (distinct count) data types.
hllppcount	Returns the estimated count of distinct values of the results on each group using the HyperLogLog++ algorithm: <input type="checkbox"/> <input checked="" type="checkbox"/> <code>hllppcount(column)</code> <input type="checkbox"/> <input checked="" type="radio"/> Applies on float or integer data types.



By **Rafa Hernández (elpluto)**
cheatography.com/elpluto/

Published 2nd January, 2023.
Last updated 2nd January, 2023.
Page 2 of 8.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
<https://apollopad.com>

String operators and functions

has, [->] Case sensitive **contains** comparison. Using the operator
-> only allows check one value:

- `has (column, value1, [value2],...)`
- `column -> value1`

weakhas Case insensitive **contains** comparison:
`weakhas (column, value)`

in, [<-] Case sensitive **is contained** comparison. Using the operator '<-' allows only one value:

- `in (value1, [value 2], [...], column)`
- `value1 <- column`

weakin Case insensitive **is contained** comparison:
`weakin (value, column)`

startswith Returns strings that start with specific value:
`startswith (column, value)`

endswith Returns strings that end with a specific value:
`endswith (column, value)`

toktains Specialized **contains** function for ASCII delimited tokens:
`toktains (column, value, [bool_ left], [bool_ right])`

length Returns the length of a string value:
`length (column)`

locate Returns the position of a substring, **indexOf** function:
`locate (column, substring_to Locate)`

lower Returns the transformation to lower case:
`lower (column)`

upper Returns the transformation to upper case:
`upper (column)`

String operators and functions (cont)

replace Replaces **only first occurrence** of a string with a substitute string:
`replace (column, string ToS earch, string ToR eplace)`

replaceall Replaces **all occurrences** of a search string with a substitute string:
`replaceall (column, string ToS earch, string ToR eplace)`

split Returns a specific piece of splitting operation by a separator:
 pieceN umber begin at 0.
`split (column, separator String, pieceN umber)`

splitre Returns a specific piece of splitting operation by a regular expression:
 pieceN umber begin at 0.
`splitre (column, re (string) or regexp, pieceN umber)`

substring Returns a substring beginning at specific index with the provided length:
`substring (column, index, length)`

subs Returns a string replacing **first substring occurrence** based on a regular expression using a template string as substitution value:
 FailValue is returned when is provided and no occurrences found
 `subs (column, regexp, template, [failValue])`
 `subs (column, re (string), template (string), [failV alue])`



By **Rafa Hernández** (elpluto)
cheatography.com/elpluto/

Published 2nd January, 2023.
Last updated 2nd January, 2023.
Page 3 of 8.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
<https://apollopadd.com>

String operators and functions (cont)

subsall Returns a string replacing **all substring occurrences** based on a regular expression using a template string as substitution value:

- 👁 FailValue is returned when is provided and no occurrences found
- ✔ `subs(column, regexp, template, [failValue])`
- ✔ `subs(column, re(string), template(string), [failValue])`

trim Returns the result of trimming **both sides**:
`trim(column)`

ltrim Returns the result of trimming **left side**:
`ltrim(column)`

rtrim Returns the result of trimming **right side**:
`rtrim(column)`

matches, [-] Matches function that finds occurrences in a column using a regular expression:

- ✔ `matches(column, re(string) or regexp value)`
- ✔ `column ~ re(string) or regexp value`

peek Returns the part of a string based on a regular expression, optionally indicating a specific part occurrence:

- 👁 If no partNumber is provided then returns first part occurrence.

`peek(column, re(string) or regexp, [partNumber])`

formatnumber Format a number with a specific mask and locale:
`format number (numberColumn, mask, locale)`

- 👁 `format number (total Amount, "###.## ", "en-GB")`

String operators and functions (cont)

damerau Returns **Damerau** distance:
`damerau(column, value)`

hamming Returns **Hamming** distance:
`hamming(column, value)`

levenshtein Returns **Levenshtein** distance:
`levenshtein(column, value)`

osa Returns **osa** distance:
`osa(column, value)`

publicsuffix Returns the main public suffix of a hostname:
`publicsuffix(hostnameColumn)`

- 👁 `'www.my.site.co.uk' = 'co.uk'`

rootdomain Returns the root domain of a hostname part of an url:
`rootdomain(hostnameUrlColumn)`

- 👁 `'www.my.site.com' = 'site'`

rootprefix Returns the root prefix of a hostname part of an url:
`rootprefix(hostnameUrlColumn)`

- 👁 `'www.my.site.com' = 'www.my.site'`

rootsuffix Returns the root suffix of a hostname part of an url:
`rootsuffix(hostnameUrlColumn)`

- 👁 `'www.my.site.com' = 'my.site.com'`

subdomain Returns the subdomain of a hostname part of an url:
`subdomain(hostnameUrlColumn)`

- 👁 `'www.my.site.com' = 'www'`

topleveldomain Returns the top level domain of a hostname part of an url:
`topleveldomain(hostnameUrlColumn)`

- 👁 `'www.my.site.co.uk' = 'uk'`

shannonentropy



By **Rafa Hernández** (elpluto)
cheatography.com/elpluto/

Published 2nd January, 2023.
Last updated 2nd January, 2023.
Page 4 of 8.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
<https://apollopad.com>

Cryptographic functions

md5
sha1
sha256
sha512

Web functions

urischeme
urihost
uriport
uripath
urifragment
uriquery
uriuser
urissp
uriauthority
absoluteuri
opaqueuri
urldecode
uaurl
uaname
uatype
uaversion
uaicon
uarobot
uainfourl
uafamily
uacompany
uacompanyurl
uadeviceicon
uadeviceinfourl
uadevicetype
uaosurl
uaosname
uaosicon
uaosfamily
uaoscompany
uaoscompanyurl
uaosversion Possible **missing** function to filter by OS version.

Meta functions

pragmavalue
tablename

Data Types

str String

int Integer number: 1, 58, 12598

float Floating point number: 24.256

boolean Boolean: true, false

timestamp Timestamp date in format: yyyy-MM-dd HH:mm:ss.SSS

boxar(int) Byte array in hexadecimal string format

duration Amount of time: an integer following by a letter [d]ays, [h]ours, [m]inutes, [s]econds, [No suffix]:m ill ise conds

geocord Geographic coordinates set:
 Latitude/ longitude sexage simal values: 40°24'N 3°41'W
 Hash representation of coordinates (geohash)

ip IPv4 address format: 192.168.5.56

ip6 IPv6 address format: 2001:0db8:85a3:0000:0000:8a2e:0370:7334

net4 IPv4 address in format: {x.x.x.x/0}

net6 IPv6 address in format: x.x.x.x.x.x/s

regexp Regular expression: [^\w]

template Represents a substitution string mask.

dc Represents a estimated count of distinct elements in a data stream.

image Image as Base64 encoding image.

mac MAC address in format: 00:0a:95:9d:68:16

namepattern Represents a part of a table name: my.app, demo, ...

set(name) Represents a set of table names: {my.app, p.test, my.app.test2}

json String in json format: {"id":345, "name":"John"}




By **Rafa Hernández** (elpluto)
cheatography.com/elpluto/

Published 2nd January, 2023.
 Last updated 2nd January, 2023.
 Page 5 of 8.

Sponsored by **ApolloPad.com**
 Everyone has a novel in them. Finish Yours!
<https://apollopad.com>

Data Types (cont)

jq Represents a **jq** filter, **jq** is a command line json processor.
 `.email`

Common comparison functions and operators

eq, [=] **Equals to** function and operator:
 `eq(column, value or column)`
 `column1 = value or column`

eqic Case insensitive **Equals to** function:
`eqic(column, value or column)`

ge, [>=] **Greater or equal** function and operator:
 `ge(column, value or column)`
 `column >= value or column`

gt, [>] **Greater than** function and operator:
 `gt(column, value or column)`
 `column > value or column`

le, [<=] **Less or equal** function and operator:
 `le(column, value or column)`
 `column <= value or column`

lt, [<] **Less than** function and operator:
 `lt(column, value or column)`
 `column < value or column`

ne, [!=] **Not equal** function and operator:
 `ne(column, value or column)`
 `column /= value or column`

isnull Check if **is null** function:
`isnull (column)`

isnotnull Check if **is not null** function:
`isnotnull (column)`

Logig Functions

and

or

not

JSON related functions

jqeval

label

jsonparse

Math functions and operators

abs

add / [+]

sub / [-]

mul / [*]

div / [/]

rdiv / [/] Real division function and operator:

mod / [%%] Module function:

rem / [%] Return the remain of a division operation:

pow Power function:

cbrt Cube root function:

sqrt Square root function:

ceil

floor

round

signum

Statistical and specialised statistical functions

estimation

pack

unpackllpp

Network functions

ispublic

isprivate

ipip4

ipprotocol

purpose

host

routing

httpstatusdescription

httpstatustype

reputation

score

sbl



By **Rafa Hernández (elpluto)**
cheatography.com/elpluto/

Published 2nd January, 2023.
 Last updated 2nd January, 2023.
 Page 6 of 8.

Sponsored by **ApolloPad.com**
 Everyone has a novel in them. Finish Yours!
<https://apollopad.com>

Conversion functions

int

str

bool

float

image

ip4

net4

ip6

net6

mac

to16

from16

to64

from64

toutf8

fromutf8

toz85

fromz85

compatible

mapped

translated

template

timestamp

duration

re

parsedate

formatdate

humansize

mkboxar

Special comparison functions

matches Matches function finds occurrences in a column using a regular expression:

✔ `matches(column, re(string) or regexp value)`

✔ `column ~ re(string) or regexp value`

Special comparison functions (cont)

anymatches Find occurrences in a **set of names** type column against a **namepattern**:

`anymatches(s(et) ofNames, nameglob(string) or namepattern)`

👁 `anymatches(t(ables), nameglob("my.app.*.*"))`

nameglob Return a formatted string as a **namepattern** to use with **anymatches**:

`nameglob(string)`

Date and time functions

day

dayofweek

dayofyear

month

year

epoch

hour

minute

second

millisecond

today

tomorrow

yesterday

period

Packet functions

hasio4

hastcp

hasudp

hasether

ip4proto

ip4src

ip4dst

ip4status

ip4ttl

ip4len

ip4payload

ip4flags



By **Rafa Hernández** (elpluto)
cheatography.com/elpluto/

Published 2nd January, 2023.
Last updated 2nd January, 2023.
Page 7 of 8.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
<https://apollopad.com>

Packet functions (cont)

ip4fragment

ip4cs

ip4hl

ip4ds

ip4ecn

ip4tos

etherdst

ethersrc

etherpayload

etherstatus

ethertag

ethertype

tcpdst

tcpsrc

tcpstatus

tcpflags

tcppack

tcpcs

tcpseq

tcpnl

tcppayload

tcpurg

tcpwin

udpsrc

udpport

udpstatus

udpcs

udplen

udppayload



By **Rafa Hernández** (elpluto)
cheatography.com/elpluto/

Published 2nd January, 2023.
Last updated 2nd January, 2023.
Page 8 of 8.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish
Yours!
<https://apollopad.com>