# Devo LINQ, query language syntax Cheat Sheet

by Rafa Hernández (elpluto) via cheatography.com/46990/cs/21055/

## Structure keywords

**from** — Point to a table for the query. Specify a table name using `from`

```
from my.app.we b.auth
```

**where** — Where clause to filter results.

```
where [filter1 expres sion], [filter2 expression]
String values in expres sions have to be surrounded by double quotes, single quotes are not
```

**select** — Add column to result set:

```
select source _column or column operation as destination_column
select uriHos t(uri) as host
```

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

```
group [every server _pe riod] [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: sec onds, nds]
every 0 means no client period.
every 5m by serverIp
```

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

```
ifthenelse(condi tion, errorV alue, succes sValue )
select ifthenelse(statu sCode != 200,"Er ror " ,"Su cce ss") as status Cod eDesc
```

## Structure keywords (cont)

**decode** — **switch/case** equivalent clause to set conditionally column valu

```
decode(column, checkV alue, value, [ch eck
Each pair of arguments [check Value, value
decode(statu sCode, 200, " Suc ces s", 400
od eDesc
```

**nvl** — **Null-Coalescing** operator. Allow to set an alternate value whe

```
nvl(column, alternate_value_when_null)`
```

## Aggregation functions and operators

**avg** — Returns the **average** of a range of values [**avg**] or only ov 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 e
```

**first / nnfirst** — Returns the **first** or the **not null first** entry of the results on

```
first(column); nnfirst(column)
```



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

Published 2nd January, 2023.
Last updated 2nd January, 2023.


Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

## Aggregation functions and operators (cont)

| | |
|---|---|
| **last / nnlast** | Returns the **last** or the **not null last** entry of the results o each group:<br>✔ **last**(column); **nnlast**(column) |
| **max / min** | Returns the **maximum** or the **minimum** value for the columns provided, on each group:<br>✔ **max/min**(col1, [col2], [col3]…) |
| **median** | Returns the statistical **median** for a column on each group:<br>✔ **median**(column)<br>👁 Restricted to columns of integer type |
| **sum** | Returns the **sum** of the results on each group:<br>✔ **sum**(column) |
| **sum2** | Returns the **sum** of the squares of the results on each group:<br>✔ **sum2**(column) |
| **percentile5**<br>**percentile10**<br>**percentile25**<br>**percentile75**<br>**percentile90**<br>**percentile95** | Returns the specific statistic **percentileN**, using linear interpolation, of the results on each group:<br>✔ **percen tile[N]**(column) |
| **stddev / nnstddev** | Returns the **biased standard deviation [stddev]** of the values or not null values **[nnstddev]** of the results on each group:<br>✔ **stddev /nn stddev**(column)<br>👁 **Biased** |

## Aggregation functions and operators (cont)

| | |
|---|---|
| **ustddev / nnustddev** | Returns the **unbiased standard deviation [ustdde** values or not null values **[unnstddev]** of the resu group:<br>✔ **ustdde v/u nns tddev**(column)<br>👁 **Unbiased** |
| **var / nnvar** | Returns the **biased variance [var]** of the values o values **[nnvar]** of the results on each group:<br>✔ **var/nnvar**(column)<br>👁 **Biased** |
| **uvar / nnuvar** | Returns the **unbiased variance [uvar]** of the valu values **[unnvar]** of the results on each group:<br>✔ **uvar/u nnvar**(column)<br>👁 **Unbiased** |
| **hllpp** | Returns the estimated count of distinct values of each group using the **HyperLogLog++** algorithm:<br>✔ **hllpp**(column)<br>👁 Applies on **DC (distinct count)** da |
| **hllppcount** | Returns the estimated count of distinct values of each group using the **HyperLogLog++** algorithm:<br>✔ **hllppcount**(column)<br>👁 Applies on **float or integer** data |



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

Published 2nd January, 2023.
Last updated 2nd January, 2023.


Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

# Devo LINQ, query language syntax Cheat Sheet

by Rafa Hernández (elpluto) via cheatography.com/46990/cs/21055/

## 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:

```
weakha s(c olumn, 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:

```
starts wit h(c olumn, value)
```

**endswith** Returns strings that end with a specific value:

```
endswi th( column, value)
```

**toktains** Specialized **contains** function for ASCII delimited tokens:

```
toktai ns( column, value, [bool_ left], [bool_ rig
ht])
```

**length** Returns the length of a string value:

```
length (co lumn)
```

**locate** Returns the position of a substring, **indexOf** function:

```
locate (co lumn, substr ing _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

```
replac e(c olumn, string ToS earch, string
```

**replaceall** Replaces **all occurrences** of a search string with a substitut

```
replac eal l(c olumn, string ToS earch, str
```

**split** Returns a specific piece of splitting operation by a separat

- `pieceN umber begin at 0.`

```
split( column, separa tor String, pieceN u
```

**splitre** Returns a specific piece of splitting operation by a regular e

- `pieceN umber begin at 0.`

```
splitr e(c olumn, re(string) or regexp, pi
```

**substring** Returns a substring beginning at specific index with the pro

```
substr ing (co lumn, index, length)
```

**subs** Returns a string replacing **first substring occurrence** based template string as substitution value:

- `FailValue is returned when is provided a`

```
d
```

- `subs(c olumn, regexp, template, [failVal`
- `subs(c olumn, re(str ing), templa te( st`



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

Published 2nd January, 2023.
Last updated 2nd January, 2023.


Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

## String operators and functions (cont)

| | |
|---|---|
| **subsall** | Returns a string replacing **all substring occurrences** based on a regular expression. Support template string as substitution value:<br>👁 `FailValue is returned when is provided and no occurr ences found.`<br>✔ `subs(c olumn, regexp, template, [failValue])`<br>✔ `subs(c olumn, re(str ing), templa te( str ing), [failV alue])` |
| **trim** | Returns the result of trimming **both sides**:<br>`trim(c olumn)` |
| **ltrim** | Returns the result of trimming **left side**:<br>`ltrim( column)` |
| **rtrim** | Returns the result of trimming **right side**:<br>`rtrim( column)` |
| **matches, [~]** | Matches function that finds occurrences in a column using a regular expression:<br>✔`matche s(c olumn, re(string) or regexp value)`<br>✔`column ~ re(string) or regexp value` |
| **peek** | Returns the part of a string based on a regular expression, optionally indicating a specific part occurrence:<br>👁 `If no partNumber is provided then returns first part occurrence.`<br>`peek(column, re(string) or regexp, [partN umber])` |
| **formatnumber** | Format a number with a specific mask and locale:<br>`format num ber (nu mbe rCo lumn, mask, locale)`<br>👁`format num ber (to tal Amount, " ###.## ", " en- GB")` |

## String operators and functions (cont)

| | |
|---|---|
| **damerau** | Returns **Damerau** distance:<br>`damera u(c olumn, value)` |
| **hamming** | Returns **Hamming** distance:<br>`hammin g(c olumn, value)` |
| **levenshtein** | Returns **Levenstein** distance:<br>`levens hte in( column, value)` |
| **osa** | Returns **osa** distance:<br>`osa(co lumn, value)` |
| **publicsuffix** | Returns the main public suffix of a hostname:<br>`publicsuffix(hostnameColumn))`<br>👁 `'www.m y.s ite.co.uk' = 'co.uk'` |
| **rootdomain** | Returns the root domain of a hostname part of an url:<br>`rootdomain(hostnameUrlColumn)`<br>👁 `'www.m y.s ite.com' = 'site'` |
| **rootprefix** | Returns the root prefix of a hostname part of an url:<br>`rootprefix(hostnameUrlColumn))`<br>👁 `'www.m y.s ite.com' = 'www.m y.si te'` |
| **rootsuffix** | Returns the root suffix of a hostname part of an url:<br>`rootsuffix(hostnameUrlColumn))`<br>👁 `'www.m y.s ite.com' = 'my.si te.c om'` |
| **subdomain** | Returns the subdomain of a hostname part of an url:<br>`subdomain(hostnameUrlColumn)`<br>👁 `'www.m y.s ite.com' = 'www'` |
| **topleveldomain** | Returns the top level domain of a hostname part of an url::<br>`topleveldomain(hostnameUrlColumn)`<br>👁 `'www.m y.s ite.co.uk' = 'uk'` |
| **shannonentropy** | |



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

Published 2nd January, 2023.
Last updated 2nd January, 2023.


Sponsored by **Readable.com**
Measure your website readability!
https://readable.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.` |
| **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, [` `onds` |
| **geocord** | Geographic coordinates set: ✔ `Latitu de/ lon gitude sexage simal val` ✔ `Hash repres ent ation of coordi nates` |
| **ip** | IPv4 address format: 👁 `192.16 8.5.56` |
| **ip6** | IPv6 address format: 👁 `2001:0 db8 :85 a3: 000 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 |
| **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.ap p.test` |
| **json** | String in json format: 👁 `{"id " :345, " nam e":` |



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

Published 2nd January, 2023.
Last updated 2nd January, 2023.


Sponsored by **Readable.com**
Measure your website readability!
https://readable.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(c olumn, 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 (co lumn)
```

**isnotnull** Check if **is not null** function:

```
isnotn ull (co lumn)
```

## 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**

**unpackhllpp**

## Network functions

**ispublic**

**isprivate**

**ipip4**

**ipprotocol**

**purpose**

**host**

**routing**

**httpstatusdescription**

**httpstatustype**

**reputation**

**score**

**sbl**



Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

# Devo LINQ, query language syntax Cheat Sheet

by Rafa Hernández (elpluto) via cheatography.com/46990/cs/21055/

## 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:

✔ `matche s(c olumn, re(string) or regexp valu e)`

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

## Special comparison functions (cont)

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

`anymat che s(s etO fNames, namegl ob( str: ttern)`

👁 `anymat che s(t ables, namegl ob( " my.a`

**nameglob** Return a formmated string as a **namepattern** to use with

`namegl ob( 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.


Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

| Packet functions (cont) |
| --- |
| ip4fragment |
| ip4cs |
| ip4hl |
| ip4ds |
| ip4ecn |
| ip4tos |
| etherdst |
| ethersrc |
| etherpayload |
| etherstatus |
| ethertag |
| ethertype |
| tcpdst |
| tcpsrc |
| tcpstatus |
| tcpflags |
| tcppack |
| tcpcs |
| tcpseq |
| tcphl |
| 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.


Sponsored by **Readable.com**
Measure your website readability!
https://readable.com