Power Query (M language) cheat sheet

**Note: M is a case sensitive language!**

|  |  |  |
| --- | --- | --- |
| **Kind** | **Literal** | **Comment** |
| Null | null | Empty value, void  1 \* null = null // be careful! |
| Logical | true / false |  |
| Number | 0 1 -1 1.5 2.3e-5, 0xff | Whole / decimal number, number in hex |
| Time | #time(9, 15, 0) | #time( hour, minute, second )  #time(24,0,0) = #time(0,0,0)  If hour is 24, then minute and second must be 0  0 ≤ hour ≤ 24, 0 ≤ minute ≤ 59, 0 ≤ second ≤ 59 |
| Date | #date(2013, 2, 26) | #date( year, month, day) |
| DateTime | #datetime(2013, 2, 26, 9, 15, 0) | #datetime( year, month, day, hour, minute, second ) |
| DateTimeZone | #datetimezone(2013, 2, 26, 9, 15, 0, 9, 0) | #datetimezone( year, month, day, hour, minute, second, offset-hours, offset-minutes )  0 ≤ year ≤ 9999, 0 ≤ month ≤ 12, 1 ≤ day ≤ 31  0 ≤ hour ≤ 23, 0 ≤ minute ≤ 59, 0 ≤ second ≤ 59  -14 ≤ offset-hours + offset-minutes / 60 ≤ 14 |
| Duration | #duration( 0, 1, 30, 0) | #duration( days, hours, minutes, seconds ) |
| Text | "hello" | Just text in quotes  Special characters  ="#(cr,lf)" same as ="#(cr)#(lf)",  string to check ="a#(cr,lf)b"  = "a#(tab)b" // a b  = "a" & "b""c" // ab"c |
| Binary | #binary("AQID") | If you work with binary – you know |
| List | { 1, 2, 3 }, { 1 .. 10 },  {"A".."Z", "a".."z"} | Comma separated values in curly brackets |
| Record | [ A=1, B=2 ] | Comma separated "Field Name = Value" in square brackets |
| Table | Simple way:  #table( { "X", "Y" }, { { 1, 2 }, { 3, 4 } } )  Preferable: with specified column types  #table( type table  [Digit ID = number, Name = text],  { {1,"one"},  {2,"two"},  {3,"three"} } ) | result:  C:\Users\bondaiva\AppData\Local\Temp\msohtmlclip1\02\clip_image001.png    #table( list of field names,  list of lists with values for rows of future table )  #table( { "Field1 Name", "Field2 Name" },  { { "Field1 Value1", "Field2 Value1" },   { "Field1 Value2", "Field2 Value2" },  { "Field1 Value3", "Field2 Value3" } } )    Empty table: #table( {"A", "B"}, {} ) |
| Function | (x) => x + 1 | ( arguments ) => some operations.  “nullable” argument is optional.  (num as nullable number) =>  let  step1 = if num = null then 0 else num,  step2 = step1 \* 2  in  step2 |
| Type | type{ number } // list  type table [A = any, B = text] |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Operator |  | x = y | Equal |
| x > y | Greater than | x<>y | Not equal |
| x >= y | Greater than or equal | x or y | Conditional logical OR |
| x < y | Less than | x and y | Conditional logical AND |
| x <= y | Less than or equal | not x | Logical NOT |

|  |  |
| --- | --- |
| **Expressions**  "Hello World" // a text value  123 // a number  1 + 2 // sum of two numbers  {1, 2, 3} // a list of three numbers  [ x = 1, y = 2 + 3 ] // a record containing two fields: x and y  (x, y) => x + y // a function that computes a sum  if 2 > 1 then 2 else 1 // a conditional expression  let x = 1 + 1 in x \* 2 // a let expression  error "A" // error with message "A" | **Recursion (**[blog post](http://www.thebiccountant.com/2017/09/26/recursion-m-beginners/)**)**  Factorial = (n) =>  if n <= 1 then  1  else  n \* **@**Factorial(n - 1),  x = Factorial(5)  // **@** is scoping operator |

**Relative dates**

Today= Date.From(DateTime.FixedLocalNow()),

Yesterday= Date.AddDays(Date.From(DateTime.FixedLocalNow()), -1),

#"End of last month" = Date.EndOfMonth(Date.AddMonths(DateTime.FixedLocalNow(), -1)),

#"Start of Current Year"= Date.StartOfYear( DateTime.FixedLocalNow() ),

#"Start of Previous Year"= Date.AddYears(Date.StartOfYear(DateTime.FixedLocalNow()), -1),

#"ISO Date format"=Date.ToText( Date.From(DateTime.FixedLocalNow()), "yyyy-MM-ddT00:00:00"),

#"Start of Month 12 months ago Excluding cur month"= Date.StartOfMonth(Date.AddMonths(DateTime.FixedLocalNow(), -12)),

#"Start of Month 12 months ago Including cur month"= Date.StartOfMonth(Date.AddMonths(DateTime.FixedLocalNow(), -11)),

// Generate Calendar – ([blog post](https://bondarenkoivan.wordpress.com/2015/10/10/generation-of-custom-calendars-in-power-query/)), [solution for Power BI](https://github.com/migueesc123/CalendarCreator4PowerBI)

// List of dates for PrevYear - Today

let

start = Date.AddYears(Date.StartOfYear(DateTime.FixedLocalNow()), -1), // start of prev year

end = Date.From(DateTime.FixedLocalNow()), // today

duration = Duration.Days(end - start) + 1,

list\_of\_dates = List.Dates(start, duration, #duration(1,0,0,0)),

#"Table from List" = Table.FromList(list\_of\_dates, Splitter.SplitByNothing(), null, null, ExtraValues.Error)

in

#"Table from List"

Get working days – try [solution from Marco Russo](https://github.com/marcosqlbi/DaxDateTemplate)

Option 1: [Parse table from TimeAndDate.com](https://www.timeanddate.com/calendar/custom.html?ctf=4&hol=9&typ=3&hod=7&holmark=1&display=2&cdt=1&wdf=4&mtm=2&cols=1&country=20&year=2017&y2=2018)

Option 2: [Use API TimeAndDate.com](https://datachant.com/2017/04/13/calendar-with-working-days-power-bi/)

Russia: [читать в блоге](https://bondarenkoivan.wordpress.com/2016/11/23/%D1%81%D0%BA%D0%B0%D1%87%D0%B0%D1%82%D1%8C-%D0%BF%D1%80%D0%BE%D0%B8%D0%B7%D0%B2%D0%BE%D0%B4%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9-%D0%BA%D0%B0%D0%BB%D0%B5%D0%BD%D0%B4%D0%B0%D1%80%D1%8C/), [function on GitHub](https://github.com/IvanBond/pquery/blob/master/List.Dates.HolidaysRU.pq)

Power Query code shortcuts

**IF / THEN / ELSE**

Result = if [Column1]>0 then [Column A] else [Column B] // low case if / then / else, M is case sensitive

**TRY / CATCH – error handling**

Result = try A/B otherwise 0 // low case “**try** [some action] **otherwise** [some action/object]”

**Excel cell value (Named Range consisting of one cell)**

Result = Excel.CurrentWorkbook(){[Name="CELLNAME"]}[Content]{0}[Column1]

**Rename Columns according to “Renaming Table”**

Renamed\_Columns = Table.RenameColumns(TARGET, Table.ToColumns(Table.Transpose(RENAMING\_TABLE)), MissingField.Ignore),

where RENAMING\_TABLE looks like

|  |  |
| --- | --- |
| Old Name | New Name |
| A | B |
| C | D |

**Rename using List.Zip, when you know order of your columns (**[blog post](https://datachant.com/2017/01/26/power-bi-pitfall-6/)**)**

Renamed\_Columns = Table.RenameColumns(TARGET,

List.Zip( { Table.ColumnNames( Source ), { "Sales Org", "Territory Key" } } ) , MissingField.Ignore),

**Create a table from thin air**

For example, when response is null but you want to keep structure of your PowerPivot table

= #table( {"A", "B"}, {} ) – empty table, simple approach

Or with defined column types

= #table( type table [A = text, B = number], {} ) – empty table

= #table( type table [My Column A = text, B = number], { {"one", 1}, {"two", 1} } )

**ISNUMBER() analog**

= Value.Is(Value.FromText( VALUE ), type number)

Or:

= "sample" is number // false, = 123 is number // true

**ISTEXT() analog**

= Value.Is(Value.FromText( VALUE ), type text)

Or:

= "sample" is text // true, = 123 is text // false

**Convert all columns of table Source to text data type**

= Table.TransformColumnTypes(Source,

List.Transform( Table.ColumnNames(Source), each { \_, type text } ) )

**Expand from nested table all not existing in current table**

= Table.ExpandTableColumn( buffer, "NewColumn",

List.Difference( Table.ColumnNames( buffer[NewColumn]{0} ), Table.ColumnNames( buffer ) ),

List.Difference( Table.ColumnNames( buffer[NewColumn]{0} ), Table.ColumnNames( buffer ) ) )

**Expand from nested table only specified in list "fields"**

Safe way to expand - it takes only intersection of Difference vs fields

  // take column Attribute from INPUT\_TABLE

fields = List.Buffer( InputTable[Attribute] ),

#"Expanded NewColumn" = Table.ExpandTableColumn( buffer, "NewColumn",

List.Intersect( { List.Difference( Table.ColumnNames( buffer[NewColumn]{0} ), Table.ColumnNames( buffer ) ), fields } ) ,

List.Intersect( { List.Difference( Table.ColumnNames( buffer[NewColumn]{0} ), Table.ColumnNames( buffer ) ), fields } ) ),

**Expand from nested table specified in special list "fields" + rename + add prefix**

fields = List.Buffer( InputTable[Attribute] ),

#"Expanded NewColumn" = Table.ExpandTableColumn( buffer, "NewColumn",

List.Intersect( { Table.ColumnNames( buffer[NewColumn]{0} ), fields } ),

// add prefix to each field

List.Transform( // rename according to RENAME\_TABLE by replacing items in list

[List.ReplaceMatchingItems](https://msdn.microsoft.com/en-us/library/mt253640.aspx)( List.Intersect( { Table.ColumnNames( buffer[NewColumn]{0} ),

fields } ),

Table.ToColumns( Table.Transpose(RENAME\_TABLE)) ),

each "Parent " & \_ )

),

**Query Folding for SQL (**[blog post](https://bondarenkoivan.wordpress.com/2016/07/17/query-folding-and-dynamic-parameters/)**)**

// Use filter as one of the first actions in Power Query after Sql.Database

// replicate “IN” clause using List.Contains

Table.SelectRows( Source, each [OrganizationKey]=11 and

List.Contains( {6,7}, [DepartmentGroupKey] ) )

**Libraries with custom Power Query functions**

<https://github.com/Hugoberry/PowerQueryExtensions> + [Hugoberry’s Gist](https://gist.github.com/Hugoberry)

<https://github.com/tycho01/pquery>

<https://github.com/tnclark8012/Power-BI-Desktop-Query-Extensions>

<https://github.com/ImkeF/RM>

<https://github.com/hohlick/PowerQueryModules>

<https://github.com/acaprojects/m-tools>

Operations with date and time in Power Query

**Time**

**#time( hour, minute, second )**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Left Operand** | **Right Operand** | **Meaning** |
| x + y | time | duration | Date offset by duration |
| x + y | duration | time | Date offset by duration |
| x - y | time | duration | Date offset by negated duration |
| x - y | time | time | Duration between dates |
| x & y | date | time | Merged datetime |

**Date**

**#date( year, month, day)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Left Operand** | **Right Operand** | **Meaning** |
| x + y | date | duration | Date offset by duration |
| x + y | duration | date | Date offset by duration |
| x - y | date | duration | Date offset by negated duration |
| x - y | date | date | Duration between dates |
| x & y | date | time | Merged datetime |

**DateTime**

**#datetime( year, month, day, hour, minute, second )**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Left Operand** | **Right Operand** | **Meaning** |
| x + y | datetime | duration | Datetime offset by duration |
| x + y | duration | datetime | Datetime offset by duration |
| x - y | datetime | duration | Datetime offset by negated duration |
| x - y | datetime | datetime | Duration between datetimes |

**Duration**

**#duration( days, hours, minutes, seconds )**

#duration(0, 0, 0, 5.5) // 5.5 seconds

#duration(0, 0, 0, -5.5) // -5.5 seconds

#duration(0, 0, 5, 30) // 5.5 minutes

#duration(0, 0, 5, -30) // 4.5 minutes

#duration(0, 24, 0, 0) // 1 day

#duration(1, 0, 0, 0) // 1 day

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Left Operand** | **Right Operand** | **Meaning** |
| x + y | datetime | duration | Datetime offset by duration |
| x + y | duration | datetime | Datetime offset by duration |
| x + y | duration | duration | Sum of durations |
| x - y | datetime | duration | Datetime offset by negated duration |
| x - y | datetime | datetime | Duration between datetimes |
| x - y | duration | duration | Difference of durations |
| x \* y | duration | number | N times a duration |
| x \* y | number | duration | N times a duration |
| x / y | duration | number | Fraction of a duration |

Main source is M language specification: <https://msdn.microsoft.com/en-us/library/mt211003.aspx>

Actual version of cheat sheet: <https://github.com/IvanBond/Power-Query-Cheat-Sheet>

Recommended blogs

<https://bondarenkoivan.wordpress.com/> - [Ivan Bondarenko](https://www.linkedin.com/in/bondarenkoivan/) ([@\_Ivan\_Bond](https://twitter.com/_ivan_bond))

<https://blog.crossjoin.co.uk/> - [Chris Webb](https://www.linkedin.com/in/chriswebb6/) ([@Technitrain](https://twitter.com/Technitrain))

<http://datachant.com/> - [Gil Raviv](https://www.linkedin.com/in/gilraviv/) ([@gilra](https://twitter.com/gilra))

<https://www.excelguru.ca/blog> - [Ken Puls](https://www.linkedin.com/in/kenpuls/) ([@kpuls](https://twitter.com/kpuls))

<https://querypower.com/> - [Igor Cotruta](https://www.linkedin.com/in/igorco/) ([@igocrete](https://twitter.com/igocrete))

<http://exceleratorbi.com.au/> - [Matt Allington](https://www.linkedin.com/in/mattallington/) ([@ExceleratorBI](https://twitter.com/ExceleratorBI))

<http://excel-inside.pro/> - [Maxim Zelensky](https://www.linkedin.com/in/maximzelensky/) ([@Hohlick](https://twitter.com/Hohlick))

<http://www.thebiccountant.com/> - [Imke Feldman](https://www.linkedin.com/in/imkefeldmann/) ([@TheBiccountant](https://twitter.com/TheBIccountant))

<https://powerpivotpro.com/> - [Rob Collie](https://www.linkedin.com/in/robcollie/), [Avi Singh](https://www.linkedin.com/in/avichalsingh/) and others ([@powerpivotpro](https://twitter.com/powerpivotpro))

In Russian:

<https://www.facebook.com/groups/Excelforever/>

<http://www.excel-vba.ru/?s=power+query>

<http://needfordata.ru/blog/>

Author – Ivan Bondarenko



<https://bondarenkoivan.wordpress.com/>

Twitter: [@\_Ivan\_Bond](https://twitter.com/_Ivan_Bond)

[Ivan Bondarenko](https://bondarenkoivan.wordpress.com/about/) - 19-Oct-2017