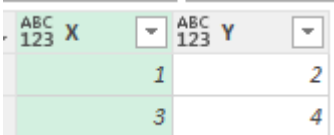


# Шпаргалка по Power Query (язык M)

Помни: M – это чувствительный к регистру язык! "А" и "а" – не одно и то же.

Тип данных	Коротко	Подробно
null	null	Пустое значение. Это не ноль и не пустая строка. 1 * null = null // будь внимателен!
logical	true / false	Истина / ложь
number	0 1 -1 1.5 2.3e-5, 0xff (hex)	Целое / действительное число
time	#time(9, 15, 0) – время Если час = 24, то минуты и секунды должны быть = 0	#time( часы, минуты, секунды ) #time(24,0,0) = #time(0,0,0) 0 ≤ часы ≤ 24, 0 ≤ минуты ≤ 59, 0 ≤ секунды ≤ 59
date	#date(2013, 2, 26)	#date( год, месяц, день ) – дата
datetime	#datetime(2013, 2, 26, 9, 15, 0)	#datetime( год, месяц, день, часы, минуты, секунды)
datetimezone	#datetimezone(2013, 2, 26, 9, 15, 0, 9, 0)	#datetimezone( год, месяц, день, часы, минуты, секунды, сдвиг часов, сдвиг минут), Дата-время с часовым поясом. Диапазоны значений: 0 ≤ год ≤ 9999, 0 ≤ месяц ≤ 12, 1 ≤ день ≤ 31 0 ≤ часы ≤ 23, 0 ≤ минуты ≤ 59, 0 ≤ секунды ≤ 59 -14 ≤ сдвиг часов + сдвиг минут / 60 ≤ 14
duration	#duration( 0, 1, 30, 0)	#duration(дни, часы, минуты, секунды) -Длительность
text	"hello" Строка с переносом ="a#(cr,lf)b" ="a" & "b""c" // ab"с – включение кавычки в строку	Текст - это просто строка в кавычках Специальные символы: перенос строки ="#(cr,lf)" тоже что и ="(cr)#(lf)", ="a#(tab)b" // a b – табуляция
binary	#binary("AQID")	Кто работает с binary, тот знает
list	{ 1, 2, 3 }, { 1 .. 10 }, {"A".."Z", "a".."z"}	list (список) это набор значений, разделенных запятой, заключенный в фигурные скобки
record	[ A=1, B=2 ] Запись может хранить несколько полей. Тип значения в записи может быть любым.	Запись: "Имя поля = значение" в квадратных скобках.
table	Простая таблица (без указания типа данных): #table( { "X", "Y" }, { { 1, 2 }, { 3, 4 } } )  Желательно указывать тип данных #table( type table [Digit ID = number, Name = text], { {1,"one"}, {2,"two"}, {3,"three"} } )  Таблица без строк: #table( {"A", "B"}, { } )	Что получится:  #table( список названий столбцов (полей), Список из списков значений для <u>строк</u> будущей таблицы)  #table( { "Столбец 1", "Столбец 2" }, { { "Столбец1 Значение1", "Столбец2 Значение1" }, { "Столбец1 Значение2", "Столбец2 Значение2" }, { "Столбец1 Значение3", "Столбец2 Значение2" } } )
function	(x) => x + 1 Функция более сложная (x as number) => let a=x+1, b=a*2 in b	( аргументы ) => тело функции.  (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]	Тип данных «Тип», позволяет получить / проверить / задать Тип каким-то данным

Оператор		x = y	Равно
x > y	Больше	x <> y	Не равно
x >= y	Больше либо равно	x or y	Условие ИЛИ
x < y	Меньше	x and y	Условие И
x <= y	Меньше либо равно	not x	Логическое отрицание НЕ

<p><b>Выражение</b></p> <p>"Кто виноват?" // текст, т.к. в кавычках</p> <p>123 // число, т.к. без кавычек</p> <p>1 + 2 // сумма чисел</p> <p>{1, 2, "3"} // список из трех элементов, "3" - текст</p> <p>[ x = 1, y = 2 + 3 ] // запись с двумя полями</p> <p>(x, y) =&gt; x + y // функция, выполняющая сложение</p> <p>if 2 &gt; 1 then 2 else 1 // выражение проверяющее условие «если / то / иначе»</p> <p><b>let</b> x = 1 + 1 <b>in</b> x * 2 // пример выражения <b>let</b></p> <p>error "A" // ошибка с сообщением "A"</p>	<p><b>Рекурсия</b> (<a href="#">blog post</a>)</p> <p>Factorial = (n) =&gt;</p> <p>    if n &lt;= 1 then</p> <p>        1</p> <p>    else</p> <p>        n * @Factorial(n - 1),</p> <p>    x = Factorial(5)</p> <p>// @ - оператор вызова исходной функции</p> <p><b>let</b> – ключевое слово, начинающее «раздел» вычислений – несколько операций. Конец этого раздела обязательно должен быть обозначен ключевым словом <b>in</b>. После <b>in</b> тоже может быть какое-то действие. Но чаще - просто ссылка на шаг перед <b>in</b>.</p>
--	---

### Относительные диапазоны дат

```

Сегодня= Date.From(DateTime.FixedLocalNow()),
Вчера= Date.AddDays(Date.From(DateTime.FixedLocalNow()), -1),
#"Конец предыдущего месяца" = Date.EndOfMonth(Date.AddMonths(DateTime.FixedLocalNow(), -1)),
#"Начало текущего года"= Date.StartOfYear( Date.From(DateTime.FixedLocalNow()) ),
#"Начало предыдущего года"= Date.AddYears(Date.StartOfYear(DateTime.FixedLocalNow()), -1),

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

#"Начало месяца 12 месяцев назад исключая текущий"=
    Date.StartOfMonth(Date.AddMonths(DateTime.FixedLocalNow(), -12)),
#"Начало месяца 12 месяцев назад включая текущий "=
    Date.StartOfMonth(Date.AddMonths(DateTime.FixedLocalNow(), -11)),

// Генерация календаря – (blog post), solution for Power BI
// Список дат в диапазоне Предыдущий год-> Сегодня
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"

Где взять рабочие дни – смотри solution from Marco Russo
Опция 1: Parse table from TimeAndDate.com
Опция 2: Use API TimeAndDate.com
Для России: читать в блоге, function on GitHub

```

# Заготовки для Power Query

## Если / То/ Иначе

Result = if [Column1]>0 then [Column A] else [Column B] // строчными if / then / else

## TRY / CATCH – аналог функции ЕСЛИОШИБКА в Excel

Result = try A/B otherwise 0 // строчными “try [опасная операция] otherwise [в случае ошибки]”

## Значение из ячейки Excel (Именованный диапазон из одной ячейки)

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

## Переименование столбцов с помощью заранее заготовленной таблицы RENAMING\_TABLE

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

где таблица RENAMING\_TABLE выглядит так (заголовки могут быть любыми)

Старое имя	Новое имя
Иск	Новосибирск
Питер	Санкт-Петербург

## Используя функцию List.Zip, когда известен порядок столбцов ([blog post](#))

Renamed\_Columns = Table.RenameColumns(TARGET,  
List.Zip( { Table.ColumnNames( Source ), { "Организация", "Магазин" } } ), MissingField.Ignore),

## Создание таблицы «из воздуха»

Например, результат запроса в веб-сервис вернул null, но в Модель Данных надо грузить нормальную таблицу, иначе всё сломается

= #table( {"A", "B"}, {} ) – пустая таблица, простой подход. Опасен тем, что не заданы типы данных!

Лучше задать тот тип данных, который Модель Данных (Power Pivot) ожидает

= #table( type table [Мой Столбец = text, Твой Столбец = number], {} ) – пустая, но типы данных заданы

= #table( type table [Столб A = text, B = number], { {"раз", 1}, {"два", 1} } ) – таблица с парой строк

## ISNUMBER() аналог

= "пример" is number // ложь; = 123 is number // истина

## ISTEXT() аналог

= "пример" is text // истина; = 123 is text // ложь

## Привести все столбцы таблицы к текстовому типу данных

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

## Развернуть из вложенной таблицы столбцы, имен которых нет в текущей таблице

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

### Развернуть из вложенной таблицы только столбцы из специального списка

```
// в качестве списка берем столбец Attribute из таблицы 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 } ) ),
```

### Развернуть столбцы из спец списка и переименовать их в процессе

```
fields = List.Buffer( InputTable[Attribute] ),
#"Expanded NewColumn" = Table.ExpandTableColumn( buffer, "NewColumn",
    List.Intersect( { Table.ColumnNames( buffer[NewColumn]{0} ), fields } ),
    // добавляем приставку Parent к имени каждого разворачиваемого столбца
    List.Transform( // переименование по таблице RENAME_TABLE путём подмены элементов списка
        List.ReplaceMatchingItems( List.Intersect( { Table.ColumnNames( buffer[NewColumn]{0} ),
            fields } ),
            Table.ToColumns( Table.Transpose(RENAME_TABLE) ) ),
        each "Parent " & _ )
    ),
```

### Работа с запросами в SQL сервер ([blog post](#))

// Старайтесь фильтровать в Power Query сразу же после операции Sql.Database, тогда фильтрация будет происходить на стороне сервера.

// чтобы сформировался запрос с условием "IN" используйте функцию List.Contains

```
Table.SelectRows( Source, each [OrganizationKey]=11 and
    List.Contains( {6,7}, [DepartmentGroupKey] ) )
```

// будет преобразовано в WHERE OrganizationKey = 11 and DepartmentGroupKey in (6, 7)

### Библиотеки с дополнительными функциями для Power Query

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

<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>

# Операции с датой и временем в Power Query

## Время

**#time( часы, минуты, секунды )**

Оператор	Слева	Справа	Результат
x + y	time	duration	Сдвиг времени
x + y	duration	time	Сдвиг времени
x - y	time	duration	Сдвиг времени
x - y	time	time	Длительность между T1 и T2
x & y	date	time	Дата + Время

## Дата

**#date( год, месяц, день )**

Оператор	Слева	Справа	Результат
x + y	date	duration	Сдвиг даты / времени
x + y	duration	date	Сдвиг даты / времени
x - y	date	duration	Сдвиг даты / времени (назад)
x - y	date	date	Длительность между датами
x & y	date	time	Дата + Время

## ДатаВремя

**#datetime( год, месяц, день, часы, минуты, секунды )**

Оператор	Слева	Справа	Результат
x + y	datetime	duration	Сдвиг даты / времени
x + y	duration	datetime	Сдвиг даты / времени
x - y	datetime	duration	Сдвиг даты / времени (назад)
x - y	datetime	datetime	Длительность между двумя моментами времени

## Длительность

**#duration( дни, часы, минуты, секунды)**

#duration(0, 0, 0, 5.5) // 5.5 секунд

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

#duration(0, 0, 5, 30) // 5.5 минут

#duration(0, 0, 5, -30) // 4.5 минут

#duration(0, 24, 0, 0) // 1 день

#duration(1, 0, 0, 0) // 1 день

Оператор	Слева	Справа	Результат
x + y	datetime	duration	Сдвиг даты / времени
x + y	duration	datetime	Сдвиг даты / времени
x + y	duration	duration	Новая длительность
x - y	datetime	duration	Сдвиг даты / времени (назад)
x - y	datetime	datetime	Длительность между датами
x - y	duration	duration	Разница в длительности
x * y	duration	number	N раз длительность
x * y	number	duration	N раз длительность
x / y	duration	number	Доля длительности

## Рекомендуемые блоги

<https://bondarenkoivan.wordpress.com/> - Ivan Bondarenko (@ Ivan Bond)

<https://blog.crossjoin.co.uk/> - Chris Webb (@Technitrain)

<http://datachant.com/> - Gil Raviv (@gilra)

<https://www.excelguru.ca/blog> - Ken Puls (@kpuls)

<https://querypower.com/> - Igor Cotruta (@igocrete)

<http://exceleatorbi.com.au/> - Matt Allington (@ExceleatorBI)

<http://excel-inside.pro/> - Maxim Zelensky (@Hohlick)

<http://www.thebiccountant.com/> - Imke Feldman (@TheBiccountant)

<https://powerpivotpro.com/> - Rob Collie, Avi Singh and others (@powerpivotpro)

На русском:

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

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

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

<https://bondarenkoivan.wordpress.com/> - Ivan Bondarenko (@ Ivan Bond)

## Автор – Иван Бондаренко



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

Twitter: @ Ivan Bond

[Ivan Bondarenko](#) - 19-Oct-2017