

《kdb+中文教程》第二章 数据类型

Original kdbcn kdb中文教程 2020年12月4日 05:30



提示：(1) 建议按本书章节顺序阅读；(2) 本书含有代码、表格等，在PC上阅读可获得更好体验。

第二章 数据类型

本章将对q语言的基本数据类型及类型转换进行介绍。

第一节 概述

q中所有数据均由原子（atom）构成，原子是q中不可再分割的最基本的数据类型。q的基本数据类型与传统编程语言类似（见表2.1），同时还具有丰富的时间相关数据类型，便于时间序列运算。

表2.1 q与其他语言数据类型对比

q	SQL	Java	.Net
boolean	boolean	boolean	boolean
guid		UUID	GUID

byte	byte	byte	byte
short	smallint	short	int16
int	int	integer	int32
long	bigint	long	int64
real	real	float	single
float	float	double	double
char	char(1)	character	char
symbol	varchar	string	(string)
timestamp		timestamp	DateTime
month			
date	Date	Date	Date
datetime	timestamp	timestamp	DateTime
timespan		timespan	timespan
minute			
second			
time	time	time	timespan

q中的数据类型有多种表示方式：类型符号（type symbol）、类型字符（type char）、类型后缀（trailing type indicator）、类型代码（type number）。其中，类型符号是由反引号`和类型名称构成的symbol，类型字符为一个代表数据类型的大写字母，类型代码则为代表该数据类型的短整型，类型后缀则是与类型字符相对应的小写字母，如下表所示。

表2.2 q的数据类型

我们可以借助type函数获取该数据的类型代码，原子型数据的类型代码为负数，简单数组（数组内容见第三章）的类型代码为正数。需要注意的是，q语言是动态类型语言，变量的类型会随着赋值数据类型的变化而变化。

例：

q)a:73 /给变量a赋值73

q)type a /a是一个长整型数据，类型代码为-7h，type函数用于显示类型代码
-7h

q)type 1 2 3 / 1 2 3是简单数组，类型代码为7h
7h

q)a:"xyz" /给变量a赋值xyz

q)type a /重新赋值后，a是一个char简单数组，类型代码为10h
10h

第二节 数值数据

本节主要介绍整数与浮点数两种数值数据。

一、整数

整数数据类型包含长整型（long）、短整型（short）、整型（int）三种类型，一般使用long类型。

（一）long

在q语言3.0版本以后，一个整数数字默认的数据类型是长整型。长整型的大小为8字节，类型代码为7h，我们一般会在数字后加上类型后缀j，但由于长整型是默认数据类型所以后缀并非强制要求。

例：

q)type 1j

-7h

q)type 1

-7h

q)type 1 3 5 /长整型简单数组

7h

q)a:56 /给变量a赋值56

q)type a /a的类型代码为-7h，是一个长整型

-7h

（二）short和int

顾名思义，短整型（short）在三种整数类型中所占内存最小，为2字节，类型代码为5h，注意必须在数字后加上类型后缀h。我们之前提到，q语言中类型代码就是用短整型来表示的。因

此，当需要进行数据类型转换时，短整型可以用来决定目标类型，在本章最后我们会详细介绍进行数据类型转换的多种方式。

例：

```
q)type 1h
```

```
-5h
```

```
q)7h$2i /int转换为long
```

```
2
```

```
q)9h$2 /long转换为float
```

```
2f
```

整型（int）大小介于长整型和短整型之间，为4字节，类型后缀为i。需要注意的是，在q语言2.8及之前的版本中，整数默认的类型为int，由于q语言的默认数据类型在不同版本间会产生变化，为保证兼容性，建议尽量明确定义数据类型。

例：

```
q)type -1i
```

```
-6h
```

需要注意的是，q语言在算数运算中会自动提升类型，例如布尔值可以参与数值计算，但在具有相同“宽”类型（字节数大）的数组中更新或添加“窄”类型（字节数小）的元素时，“窄”类型数据无法进行类型提升而导致出错。

例：

```
q)type 1j+2i
```

```
-7h
```

```
q)L:1 2 3f
```

```
q)L[2]:4j
```

```
'type
```

二、浮点数

浮点数分为双精度（float）和单精度（real）两种类型。一般使用float类型。

（一）float

双精度浮点数（float）相当于其他编程语言中的double，大小为8字节，类型后缀为f，类型代码为9h。对于小数点后为0的情况，输入后会自动省略0并加上后缀f。此外，float也可以用科学记数法表示，注意不要将“e”与类型后缀混淆，e后面数字前的正号与零可省略。例：

```
q)type 111.222
```

```
-9h
```

```
q)type 1.2 2.3 3.4 /float数组
```

```
9h
```

```
q)2.0
```

```
2f
```

```
q)type 1e9
```

```
-9h
```

```
q)1e-9
```

```
1e-009
```

（二）real

单精度浮点数（real）大小为4字节，类型后缀为e，类型代码为8h。real的精度至少可以保留6位小数，float的精度至少可以保留15位小数，因此在金融计算中，推荐使用精度较高的float。

（三）浮点数显示

需要注意的是，q控制台默认显示7位小数精度的数字。因此，我们可能无法显示太长的浮点数。要想显示全部，可以利用\p（P为大写字母）来控制显示精度。\p后面的数字表示希望显示多少位数字，不包括小数点，但包括小数点前的数字。例：

```
q) 3.1415926535
```

```
3.141593
```

```
q) \P 11
```

```
q) 3.1415926535
```

```
3.1415926535
```

第三节 文本数据

本节主要介绍两种文本数据类型。q语言有字符（char）和符号（symbol）两种原子文本类型，分别类似SQL中的CHAR和VARCHAR。

一、char

字符类型（char）用1字节来储存一个单独的ASCII码，对应于SQL中的CHAR，用双引号表示。在q语言中，双引号（double-quote）、反斜线（back-slash）等特殊字符不能直接放在双引号中成为字符，需要在其前使用转义符（\）进行转义，尽管在q控制台中依旧将\显示了出来，但是实际上仍为单字符。此外，由字符组成的数组称为字符串（string）。例：

```
q)type "x"
```

```
-10h
```

```
q)type "xyz"x"
```

```
-10h
```

```
q)"\" / 双引号
```

```
"\"
```

```
q)"\" / 反斜杠
```

```
"\"
```

```
q)"n" / 换行
```

```
"n"
```

```
q)"r" / 回车
```

```
"r"
```

```
q)"\t" / tab制表符
```

```
"\t"
```

```
q)"\142" /显示ASCII码对应的字母，142为字母b的ASCII码八进制表示
```

```
"b"
```

二、symbol

符号（symbol）数据类型用反引号（```）表示，我们在第一节提到的类型符号就是symbol类型，所以都带有反引号（如``long`，``char`）。q中所有名称均为符号，但并非所有符号均为名称。符号类似于SQL中的VARCHAR，可以容纳任意数量的字符，不同之处在于symbol与char一样，是不可再拆分的原子类型，这意味着不可直接访问组成该符号的各个字符。需要注意的是，symbol不同于string，string不是原子类型，而是字符构成的数组。并且symbol数据``a`与char数据`"a"`不相等。例：

```
q)type `z
```

```
-11h
```

```
q)type `x`y`z
```

```
11h
```

第四节 二进制数据

本节主要介绍布尔值（boolean）、字节（byte）与全局唯一标识符（GUID）三种二进制数据。

一、boolean

布尔值类型（boolean）用1字节来存储一个二进制位（bit，0或1），注意需要加上类型后缀b。布尔值可参与计算并自动进行类型提升，即运算结果的类型为与其进行运算的数据类型相同，这个特性使得条件可以参与运算。

例：

q) type 0b
-1h
q) type 1b
-1h
q) type 10b /boolean数组，如11b,101b,111b等
1h
q) a: 1+0b
q) type a
-7h
q) flag:1b
q) base:10
q) base+flag*6
16

二、byte

字节（byte）数据类型大小为1字节，以0x为前缀，后面是两个16进制数字，可以使用大写或小写字母作为字母的十六进制数字，但通常使用小写字母。类型代码为4h。与布尔值数据一样，byte也可参与计算并自动进行类型提升。例：

q) type 0x2a
-4h
q) type 0x2A
-4h
q) type 0x2A3B4C /字节数组
4h
q) 1+0x2a
42

三、GUID

全局唯一标识符（GUID）是在q语言3.0版本中引入的。GUID的全称是globally unique identifier，是一个16字节的二进制值，该值在时间和空间上几乎是唯一的。GUID特别适合在无需借助中央控制机制（例如交易ID）的情况下，在本地生成全局唯一标识符。GUID可以用作表的主键或在表连接中使用，在这种情况下，优于使用字符串或符号。

我们可以对0Ng（null guid）使用?来生成一个guid数组,其中正号使用同一个初始随机种子生产数组,而负号使用的种子全部是随机的。

例：

q)1?0Ng

q)-1?0Ng

q)3?0Ng

q)-3? 0ng

q)0x0 sv 16?0xff /使用命令sv从一个16个byte的数组构造GUID

第五节 日期与时间数据

本节将重点介绍kdb+最具有特色的日期与时间数据类型：date、time、datetime、timestamp、timespan、month、minute、second等。

一、日期与时间数据类型

q语言最大的优势是可以有效地处理时间序列数据，q语言中主要日期与时间数据类型的格式和总结如下表，我们可以注意到这些数据都以不同形式转换为数字。接下来，我们将对每种类型进行一一介绍。

表2.3 q的时间数据类型

(一) date

日期 (date) 大小为4字节，必须写成 yyyy.mm.dd的形式。例如，2019年7月4日要用 2019.07.04来表示，而2019.7.4则是错误的写法。日期 (date) 代表从2000.01.01计数的日期数，之后的为正值，之前的则为负值。

例：

q)2000.01.01=0 /判断2000.01.01是否等于0

1b /结果为真

q)`int\$2000.02.01 /表示的累积日数可以通过强制转换得到

(二) time与timespan

时间数据类型分为time和timespan两种。其中time精确到毫秒，表示为hh:mm:ss.uuu的形式 (hh代表小时，mm代表分钟，ss代表秒，uuu代表毫秒)，代表从凌晨 (00:00) 计算的毫秒数。与date一样，time表示值同样可以通过强制转换来获取。如果毫秒级够用的话，我们就使用time数据类型。例：

q)12:34:56.789

12:34:56.789

q)12:00:00.000=12*60*60*1000

1b

若毫秒级不够用，可使用 `timespan` 类型。该类型将凌晨（00:00）后的纳秒数 (nanoseconds) 存储为长整数，表示为 `0Dhh:mm:ss.nnnnnnnnnn`。其中，开头的 `0D` 是可选的，可以不写。例：

q)12:34:56.123456789

0D12:34:56.123456789

q)12:34:56.123456 / 微秒自动转化为纳秒

0D12:34:56.123456000

(三) `datetime`与`timestamp`

日期与时间数据类型包含 `datetime` 和 `timestamp` 两种。其中，`datetime` 类型已过时，不推荐使用，该类型用大写字母 `T` 来分隔日期与时间，表示为 `yyyy.mm.ddThh:mm:ss.uuu`。例：

q)2000.01.01T12:00:00.000

—

q)2000.01.02T12:00:00.000=1.5

1b

推荐的日期与时间类型是 `timestamp` 类型（时间戳），也就是 `date` 类型和 `timespan` 类型的组合，通过大写字母 `D` 分隔日期与时间。`timestamp` 表示从 2000.01.01 以来的纳秒数，与 `date` 数据类型一样，2000.01.01 之后的为正值，之前的则为负值。例：

q)2014.11.22D17:43:40.123456789

q)`long\$2014.11.22D17:43:40.123456789

q)`date\$2014.11.22D17:43:40.123456789

q)`timespan\$2014.11.22D17:43:40.123456789

q)2014.11.22+17:43:40.123 /date + time 将得到 timestamp类型

2014.11.22D17:43:40.123000000

(四) month

月份 (month) 数据类型存储为32位带符号的整型，表示为yyyy.mm，必须加上类型后缀m, 表示从2000.01.01计数的月份数。例：

q)2000.01m=0 /判断2000.01m是否等于0

1b /结果为真

(五) minute

分钟 (minute) 数据类型存储为32位带符号的整型，表示为hh:mm，代表从凌晨(00:00)开始计数的分钟数。例：

q)00:00=0

1b

q)12:00=12*60

1b

(六) second

秒 (second) 数据类型也是32位带符号整型，表示为hh:mm:ss，代表从凌晨(00:00)计数的秒数。例：

q)00:00:00=0

1b

q)23:59:59=-1+24*60*60

1b

二、点操作符与强制转换

我们可以通过点操作符提取年份、月份、日期等。但是更推荐使用强制转换符\$，因为它适用于所有有意义的时间数据的提取和转换，且点运算符在函数内部不起作用。例：

```
q)d:2014.01.01
q)d.year
2014i
q)d.mm
q)d.dd
q)ti:12:34:56.789
q)ti.hh
12i
q)ti.mm
q)ti.ss
q)`dd$d /强制转换
1i
q)`mm$d
q)`dd$d
q)`month$d
2014.01m
```

第六节 空值与最值

本节将介绍q中的空值（Nulls）与最值（Infinitities）。

一、空值

空值（nulls）一般表示缺失数据。在C，java等语言中，空值是未分配内存空间的指针，在q中，空值与正常值同样占用内存，这是与其他编程语言不同的地方。

（一）空值的类型

表2.4 不同类型空值

类型	空值
boolean	0b
guid	0Ng(00000000-0000-0000-0000-000000000000)
byte	0x00
short	0Nh
Int	0Ni
long	0N或0Nj
real	0Ne
float	0n或0Nf
char	" "
sym	`
timestamp	0Np
month	0Nm
date	0Nd
datetime	0Nz
timespan	0Nn
minute	0Nu
second	0Nv
time	0Nt

（二）空值的判断

q使用null函数来测试一个值是不是null值，而null函数支持各种数据类型的。

例：

q)null 20

0b

```
q)null `
1b
q)null " "
1b
q)null ""
```

二、无穷

无穷值比一般的值都大。表2.5分别列出了长整型空值、正无穷和负无穷对应的实际数值，我们可以看出： $0N < -0W < \text{正常整数} < 0W$ 。需要注意的是，小写的w代表float, 大写的W代表长整数。整型无穷可以参与比较并返回正确的结果。q的哲学是任何合法的数学表达式都会得到结果而非返回错误。在数学中，0除以0是未定义的，但在q语言中，数字的除法结果总是float，正数除以0的结果为正无穷，负数除以0的结果为负无穷。

表2.5不同类型无穷值

第七节 数据类型转换

本节介绍兼容类型转换、字符串解析等数据类型转换方法。

一、数据类型转换

类型转换（Casting）是将数据由一种类型转换为另一种兼容的数据类型，转换过程中信息可能保留也可能有所丢失。使用二元运算符“\$”进行转换。操作符右边的参数是原始值，左边的参数是目标类型。我们有三种方式来指定目标类型：类型符号（type symbol）、类型字符（type char）、类型代码（type number），具体可见本章第一节。例：

q)`long\$2i /通过类型符号转换

2

q)"j"\$2i /通过类型简称转换，注意数值数据之间的类型转换要用小写字母

2

q)7h\$2i /通过类型代码转换

2

（一）短字节转长字节

由第一节我们可以知道，不同数据类型所占字节大小不同，如果由短字节（窄字节）数据类型转换为长字节（宽字节）数据类型，一般不会造成信息丢失。例：

q)7h\$2i /int转换为long

2

q)9h\$2 /long转换为float

2f

q)"j"\$2i /int转换为long

2

q)"f"\$2 /long转换为float

2f

q)`long\$2i /int 转换为long

2

q)`float\$2 /long转换为float

2f

(二) 长字节转短字节

与之对应，如果由长字节数据类型转换为短字节数据类型，则可能造成信息丢失。

1、长字节数值转化为短字节数值

例：

q)`long\$3.14159 /由float转换为long造成信息丢失

3

q)`short\$123456789 /由long转换为short造成信息丢失

32767h

2、数值数据转换为boolean

将数值类型转化为布尔值时，为0的数值转化为0b，其他数值转化为1b，例：

q)`boolean\$0

0b

q)`boolean\$123

1b

q)`boolean\$-12.345

1b

3、复杂类型中提取成分

例：

q)`date\$2015.01.02D10:20:30.123456789

2015.01.02

q)`year\$2015.01.02

2015i

q)`month\$2015.01.02

2015.01m

q)`mm\$2015.01.02

1i

q)`dd\$2015.01.02

2i

q)`hh\$10:20:30.123456789

10i

q)`minute\$10:20:30.123456789

10:20

q)`uu\$10:20:30.123456789

20i

q)`second\$10:20:30.123456789

10:20:30

q)`ss\$10:20:30.123456789

30i

（三）跨数据类型间转换

1、字符串与整数互转

在前面我们提到字符串（char）类型存储的是ASCII码，所以只要整数不超过256，我们就可以将char与整数互相转换。

例：

q)`char\$42 /ASCII码42代表的符号是*号

"*"

q)`long\$"\n" /符号\n代表的ASCII是10
10

2、日期与整数互转

例：

q)`date\$0
2000.01.01
q)`long\$2001.01.01
366

3 `timespan与长整数互转

例：

q)`long\$12:34:56.123456789
45296123456789
q)`timespan\$150000000000
0D00:02:30.000000000

（四）整数极值转换

例：

q)`int\$0Wh
32767i
q)`int\$-0Wh
-32767i
q)`long\$0Wi
2147483647
q)`long\$-0Wi

-2147483647

(五) 强制类型转换

我们对简单数组赋值时需要严格匹配数组的数据类型，这种时候可以通过强制类型转换来实现。例：

```
q)L:10 20 30 40
```

```
q)L[1]:42h
```

```
'type
```

```
q)L,:43h
```

```
'type
```

```
q)L[1]:(type L)$42h
```

```
q)L,:(type L)$43h
```

(六) 转换为原子型操作

cast是原子型函数,可以应用到左右两侧的每一个元素。对于q语言中的基础数据类型，我们可以同时对多个数据进行类型转换。例：

```
q)"i"$10 30 50
```

```
10 30 50i
```

```
q)`float$(12j; 12i; 12j)
```

```
12 12 12f
```

```
q)`short`int`long$12
```

```
12h
```

```
12i
```

```
12
```

```
q)"ijf"$98.6
```

```
99i
```

```
99
```

```
98.6
q)"ijf"$10 30 50
10i
30
50f
```

二、数据和文本的转换

（一）数据转换为string

我们之前在介绍char时提到过，q语言的string其实是由char组成的数组。我们还可以使用string函数将任何q实体转换为适合控制台展示或储存的文本形式。需要注意的是，string函数不是原子型函数，存在如下特点：（1）转换结果是一个char数组，不是单个的char；（2）转换结果不会包含任何q类型标识，可能无法解析回原始值；（3）对string数据使用string函数可能不会得到我们想要的结果。

例：

```
q)string 12 /转换后的"12"不再是12，而是包含"1"和"2"两个char元素的数组，即字符串
"12"

q)string 1 /转换后的1变成单元素数组 "1"，即元素为一个"1"字符的字符数组
,"1"

q)string 12i
"12"

q)a:3.0
q)string a
,"3"

q)string 1 3 5
,"1"
,"3"
```

```
, "5"  
q)string "test" / "test" 是一个char类型组成的数组  
, "t"  
, "e"  
, "s"  
, "t"  
q)string (1 3 5; 20 40 60)  
, "1" , "3" , "5"  
"20" "40" "60"  
q)string `Life`Is`Beautiful /将symbol转换为string  
"Life"  
"Is"  
"Beautiful"
```

(二) string转换为symbol

我们可以使用 `将string转换为symbol，这也是创建带有空格或者其它特殊符号symbol的唯一方法。

例：

```
q)`$ "xyz" /作为string类型，“ xyz”三个元素组成的数组  
`xyz /作为symbol类型，xyz是一个元素  
q)`$ "Hello World"  
`Hello World  
q)`$ ("Life"; "Is"; " Beautiful")  
`Life`Is`Beautiful  
q)`$ "Zaphod \"Z\"" /注意带有转义符的转换  
`Zaphod "Z"
```

q)string ` \$" xyz " /左右去空格

"xyz"

q)` \$" 中文"

` 中文

q)` \$" \326\320\316\304" /中文GBK编码、8进制

` 中文

(三) 字符串解析

我们可以使用\$将string转换为其他类型的数据，\$右边是要解析的字符串，\$左边是类型字符，代表希望转换成的目标数据类型。例：

q)"J"\$ "12"

12

q)"F"\$ "12"

12f

q)"F"\$ "12.0"

12f

q)"I"\$ "12.0"

0Ni

q)"I"\$ " "

0Ni

我们可以用这个方法将string日期信息转换为日期数据类型。例：

q)"D"\$ "12.31.2018"

2018.12.31

q)"D"\$ "12-31-2018"

2018.12.31

q)"D"\$ "12/31/2018"

2018.12.31

q)"D"\$"12/31/2018"

2018.12.31

q)"D"\$"2018/12/31"

2018.12.31

声明：《kdb+中文教程》版权归原作者所有，未经原作者书面允许不得转载本书除前言、第一章、第二章以外的内容，否则将视为侵权。转载本书前言、第一章、第二章或者引用本书内容请注明来源并保留完整内容。对不遵守本声明或其他违法、恶意使用本书内容者，本书作者保留追究其法律责任的权利。©版权所有 侵权必究

