

Package ‘factor256’

July 22, 2025

Title Use Raw Vectors to Minimize Memory Consumption of Factors

Version 0.1.0

Description Uses raw vectors to minimize memory consumption of categorical variables with fewer than 256 unique values. Useful for analysis of large datasets involving variables such as age, years, states, countries, or education levels.

License GPL-2

Encoding UTF-8

RoxygenNote 7.2.0

Imports utils

Suggests data.table, tinytest

NeedsCompilation yes

Author Hugh Parsonage [aut, cre]

Maintainer Hugh Parsonage <hugh.parsonage@gmail.com>

Repository CRAN

Date/Publication 2023-11-17 08:50:06 UTC

Contents

count_by256	2
factor256	2
interlace256	4
setkeyv256	5

Index	6
--------------	----------

count_by256

Aggregating helpers

Description

Aggregating helpers

Usage

```
count_by256(DT, by = NULL, count_col = "N")
```

Arguments

DT A data.table.

by (string) A column of DT, the count of which is desired.

count_col (string) The name of the column in the result containing the counts.

Value

For:

count_by256 A tally of by.

factor256

Factors of fewer than 256 elements

Description

Whereas base R's factors are based on 32-bit integer vectors, factor256 uses 8-bit raw vectors to minimize its memory footprint.

Usage

```
factor256(x, levels = NULL)
```

```
recompose256(f)
```

```
relevel256(x, levels)
```

```
## S3 method for class 'factor256'
levels(x)
```

```
is.factor256(x)
```

```
isntSorted256(x, strictly = FALSE)
```

```

as_factor(x)

factor256_in(x, tbl)

factor256_notin(x, tbl)

factor256_ein(x, tbl)

factor256_enotin(x, tbl)

tabulate256(f)

rank256(x)

order256(x)

unique256(x)

tabulate256_levels(x, nmax = NULL, dotInterval = 65535L)

```

Arguments

<code>x</code>	An atomic vector with fewer than 256 unique elements.
<code>levels</code>	An optional character vector of or representing the unique values of <code>x</code> .
<code>f</code>	A raw vector of class <code>factor256</code> .
<code>strictly</code>	If <code>TRUE</code> then if <code>x[i] == x[j]</code> and <code>i != j</code> then <code>x</code> is not sorted.
<code>tbl</code>	The table of values to lookup in <code>f</code> . May be a <code>factor256</code> class but will be implicitly converted based on the levels of <code>f</code> .
<code>nmax, dotInterval</code>	(<code>tabulate256_levels</code> only). Every <code>dotInterval</code> iterations through <code>x</code> check number of unique elements detected so far. If any count exceeds <code>nmax</code> the rest of the vector is ignored.

Value

`factor256` is a class based on raw vectors. Values in `x` absent from `levels` are mapped to `00`.

In the following list, `o` is the result.

`factor256` A raw vector of class `factor256`.

`recompose256` is the inverse operation.

`factor256_e?(not)?in` A logical vector the same length of `f`, `o[i] = TRUE` if `f[i]` is among the values of `tbl` when converted to `factor256`. `_notin` is the negation. The `factor256_e` variants will error if none of the values of `tbl` are present in `f`.

`tabulate256` Takes a raw vector and counts the number of times each element occurs within it. It is always length-256; if an element is absent it will have value zero in the output.

`tabulate256_levels` Similar to `tabulate256` but with optional arguments `nmax`, `dotInterval`.
`as_factor` Converts from `factor256` to `factor`.
`order256` Same as `order` but supports raw vectors. `order256(x)`
`rank256` Same as `rank` with `ties.method = "first"` but supports raw vectors.
`unique256` Unique elements of.

Examples

```

f10 <- factor256(1:10)

fletters <- factor256(rep(letters, 1:26))
head(factor256_in(fletters, "g"))
head(tabulate256(fletters))
head(recompose256(fletters))

gletters <- factor256(rep(letters, 1:26), levels = letters[1:25])
tail(tabulate256(gletters))
tabulate256_levels(gletters, nmax = 5L, dotInterval = 1L)
  
```

interlace256	<i>Interlace raw vectors</i>
--------------	------------------------------

Description

Some processes do not accept raw vectors so it can be necessary to convert our vectors to integers.

Usage

```

interlace256(w, x, y = NULL, z = NULL)

deinterlace256(u)

interlace256_columns(DT, new_colnames = 1L)

deinterlace256_columns(DT, new_colnames = 1L)
  
```

Arguments

<code>w, x, y, z</code>	Raw vectors. A vector may be NULL if fewer than four vectors need to be compressed.
<code>u</code>	An integer vector.
<code>DT</code>	A <code>data.frame</code> containing raw vectors to be interlaced.
<code>new_colnames</code>	A mechanism for producing the new columns. Currently only 1L is implemented, the default mechanism.

Value

interlace256 Return an integer vector, compressing raw vectors. deinterlace256 is the inverse operation, returning a list of four raw vectors.

setkeyv256	setkey <i>for raw columns</i>
------------	-------------------------------

Description

setkey for raw columns

Usage

setkeyv256(DT, cols)

Arguments

- DT A data.table.
- cols Column names as in data.table::setkeyv

Value

Same as data.table::setkeyv except that raw cols will be converted to factors (as data.table does not allow raw keys).

Index

`as_factor (factor256)`, [2](#)

`count_by256`, [2](#)

`deinterlace256 (interlace256)`, [4](#)
`deinterlace256_columns (interlace256)`, [4](#)

`factor256`, [2](#)
`factor256_ein (factor256)`, [2](#)
`factor256_enotin (factor256)`, [2](#)
`factor256_in (factor256)`, [2](#)
`factor256_notin (factor256)`, [2](#)

`interlace256`, [4](#)
`interlace256_columns (interlace256)`, [4](#)
`is.factor256 (factor256)`, [2](#)
`isntSorted256 (factor256)`, [2](#)

`levels.factor256 (factor256)`, [2](#)

`order256 (factor256)`, [2](#)

`rank256 (factor256)`, [2](#)
`recompose256 (factor256)`, [2](#)
`relevel256 (factor256)`, [2](#)

`setkeyv256`, [5](#)

`tabulate256 (factor256)`, [2](#)
`tabulate256_levels (factor256)`, [2](#)

`unique256 (factor256)`, [2](#)