

Package: Dmisc (via r-universe)

October 14, 2026

Title Daniel miscellaneous functions

Version 0.4.0

Description The Dmisc package provides a collection of versatile R functions developed by Daniel E. de la Rosa. These are general-purpose tools, not tied to any specific domain, aiming to aid a wide range of tasks in data analysis and visualization.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.1)

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

URL <https://drdsdaniel.github.io/Dmisc/>,
<https://github.com/drdsdaniel/Dmisc>

BugReports <https://github.com/drdsdaniel/Dmisc/issues>

Language en-US

RdMacros lifecycle

VignetteBuilder knitr

Imports DBI, dplyr, glue, lifecycle, magrittr, rlang, stringr, tidyr

Suggests cli, odbc, DT, flextable, forecast, ggcorrplot, ggplot2, htmltools, janitor, keyring, knitr, lmtest, lubridate, patchwork, pins, purrr, renv, RPostgres, RSQLite, rsample, rvest, scales, sf, strucchange, testthat (>= 3.2.0), tibble, tidyselect, tseries, vars, withr, xts, yardstick, zoo

Config/testthat/edition 3

Config/pak/sysreqs libicu-dev

Repository <https://adatar-do.r-universe.dev>

Date/Publication 2026-09-14 19:51:00 UTC

RemoteUrl <https://github.com/dnldelarosa/Dmisc>

RemoteRef HEAD

RemoteSha 4d9ed9aa3669c0197da2de0121992bb0535312a2

Contents

%<...%	2
%...>%	3
cut3	3
cut3_quantile	5
db_connect	6
dbConnect	7
describe	8
pin_get_or_create	9
vars_to_date	9
Index	11

%<...%	<i>Unpack values into explicitly named objects</i>
--------	--

Description

Character targets use the global environment and warn before assignment. A list of targets can include .envir and .warn options. When all requested names occur in the values, assignment matches by name and ignores extra values. Otherwise assignment is positional and lengths must match. Target validation completes before writing any object.

Usage

names %<...% values

Arguments

- | | |
|--------|---|
| names | Character target names, or a list of names with optional .envir (environment) and .warn (logical) elements. Targets must be unique. |
| values | An atomic vector or list of values. |

Value

Invisibly NULL. Objects are assigned in the selected environment.

Examples

```
output <- new.env(parent = emptyenv())
list("a", "b", .envir = output, .warn = FALSE) %<...% list(b = 20, a = 10)
output$a
```

`%...>%`

Unpack values with the values on the left

Description

Unpack values with the values on the left

Usage

values %...>% names

Arguments

values An atomic vector or list of values.
names Character targets or a list with optional .envir and .warn fields.

Value

Invisibly NULL. See `%<...%` for matching and validation rules.

Examples

```
output <- new.env(parent = emptyenv())
c(10, 20) %...>% list("a", "b", .envir = output, .warn = FALSE)
output$b
```

`cut3`

Cut a numeric column, optionally within groups

Description

Numeric boundaries apply to every group. A single number requests that many equal-width intervals within each group. Functions (or lists of numbers and functions) calculate boundaries separately for each group. Rows, other columns and existing grouping metadata are preserved.

Usage

```
cut3(
  tbl,
  var_name,
  breaks,
  groups = NULL,
  bf_args = list(),
  .inf = FALSE,
  ...,
  weights = NULL
)
```

Arguments

<code>tbl</code>	A local data frame or tibble.
<code>var_name</code>	One numeric column name.
<code>breaks</code>	Numeric boundaries, a number of intervals, a function, or a list of numbers and functions. A function must return numeric boundaries.
<code>groups</code>	Optional character vector of grouping column names. Missing group values form their own group. Existing tibble groups are not implicit.
<code>bf_args</code>	Named arguments for the boundary function, such as <code>probs</code> and <code>na.rm</code> for <code>stats::quantile()</code> .
<code>.inf</code>	Add negative and positive infinity to the boundaries. In this mode a single number is a boundary, not an interval count.
<code>...</code>	Arguments passed to <code>base::cut()</code> , including <code>labels</code> , <code>right</code> , <code>include.lowest</code> and <code>ordered_result</code> . Even <code>labels = FALSE</code> returns a factor.
<code>weights</code>	Optional weight column name or numeric vector with one value per row. Used weights must be finite and nonnegative. They affect functional boundaries only; fixed boundaries and interval counts do not read weights.

Details

With weights, `stats::quantile` uses the inverse weighted empirical distribution (type 1). Zero-weight rows do not determine boundaries. Each group needs positive total weight after missing values are removed. Other functions must explicitly accept `weights` or `w`; weights are never silently discarded. Unweighted quantiles retain the default type 7 from R. Duplicate boundaries are collapsed; labels must match the resulting intervals. Missing values remain missing. An all-missing group cannot determine automatic boundaries, but can be classified using fixed numeric boundaries.

Value

A copy of `tbl` with `var_name` replaced by a factor.

See Also

`cut3_quantile()`, `stats::quantile()`

Examples

```
x <- data.frame(value = c(1, 2, 3, 10, 20, 30), group = rep(c("A", "B"), each = 3))
cut3(x, "value", c(0, 5, 50), labels = c("low", "high"))
cut3_quantile(x, "value", .groups = "group", probs = 0.5)
```

cut3_quantile*Cut a numeric column using quantiles*

Description

Cut a numeric column using quantiles

Usage

```
cut3_quantile(  
  tbl,  
  var_name,  
  .labels = NULL,  
  .groups = NULL,  
  .inf = TRUE,  
  ...,  
  weights = NULL  
)
```

Arguments

tbl	A local data frame or tibble.
var_name	One numeric column name.
.labels	Optional labels for the distinct resulting intervals.
.groups	Optional grouping column names.
.inf	Add negative and positive infinity to the boundaries. In this mode a single number is a boundary, not an interval count.
...	Arguments to <code>stats::quantile()</code> , such as <code>probs</code> , <code>na.rm</code> and <code>type</code> . Default probabilities are 0.25, 0.5 and 0.75. Weighted quantiles use type 1 only; unweighted quantiles use R's default type 7.
weights	Optional weight column name or numeric vector with one value per row. Used weights must be finite and nonnegative. They affect functional boundaries only; fixed boundaries and interval counts do not read weights.

Value

A copy of `tbl` with `var_name` replaced by a factor.

Examples

```
x <- data.frame(value = c(1, 2, 100), weight = c(1, 8, 1))  
cut3_quantile(x, "value", weights = "weight", probs = 0.5)
```

db_connect

*Connect to PostgreSQL or SQL Server***Description**

Creates a DBI connection only when called. The caller owns the connection and must close it with `DBI::dbDisconnect()`. Explicit credentials take precedence over keyring. The historical keyring service default is "postgre"; set `k_service_id = NULL` to use explicit credentials or driver defaults.

Usage

```
db_connect(
  db_user = NULL,
  db_pass = NULL,
  db_sys = "PostgreSQL",
  db_name = "enft",
  db_host = "localhost",
  db_port = 5432,
  k_service_id = "postgre",
  ...,
  trusted_connection = "Yes"
)
```

Arguments

<code>db_user, db_pass</code>	Optional user and password. Values are passed to the driver and never printed by Dmisc.
<code>db_sys</code>	PostgreSQL or SQL Server.
<code>db_name</code>	Database name. Legacy enft1/enft2 names map to enft with a warning.
<code>db_host</code>	Server hostname.
<code>db_port</code>	Port for PostgreSQL.
<code>k_service_id</code>	Optional keyring service. A username is required if the service contains more than one account. No first-account guessing occurs.
<code>...</code>	Named driver arguments, e.g. <code>sslmode</code> for PostgreSQL or <code>Driver</code> for ODBC. Arguments already supplied by this wrapper cannot be repeated.
<code>trusted_connection</code>	TRUE/FALSE or Yes/No for SQL Server. Trusted connections do not look up passwords or send UID/PWD. Otherwise credentials are sent as UID and PWD.

Value

A DBI connection. Remote authentication is performed by the driver.

Examples

```
# A real connection is an explicit user action:
if (FALSE) {
  con <- db_connect(db_name = "example", k_service_id = NULL, sslmode = "require")
  DBI::dbDisconnect(con)
}
```

dbConnect

*Legacy database connection alias***Description**

Legacy database connection alias

Usage

```
dbConnect(
  db_user = NULL,
  db_pass = NULL,
  db_sys = "PostgreSQL",
  db_name = "enft",
  db_host = "localhost",
  db_port = 5432,
  k_service_id = "postgre",
  ...,
  trusted_connection = "Yes"
)
```

Arguments

db_user, db_pass	Optional user and password. Values are passed to the driver and never printed by Dmisc.
db_sys	PostgreSQL or SQL Server.
db_name	Database name. Legacy enft1/enft2 names map to enft with a warning.
db_host	Server hostname.
db_port	Port for PostgreSQL.
k_service_id	Optional keyring service. A username is required if the service contains more than one account. No first-account guessing occurs.
...	Named driver arguments, e.g. sslmode for PostgreSQL or Driver for ODBC. Arguments already supplied by this wrapper cannot be repeated.
trusted_connection	TRUE/FALSE or Yes/No for SQL Server. Trusted connections do not look up passwords or send UID/PWD. Otherwise credentials are sent as UID and PWD.

Value

A DBI connection. This alias emits a deprecation warning.

describe

Create a summary table for a vector or data frame

Description

Each variable is summarized separately with `base::summary()`. Statistics are aligned by name. A missing statistic for one variable is represented by NA. Columns are converted back to numeric where possible; mixed-type statistics remain character. Row and column names are preserved even for one variable.

Usage

```
describe(data, digits = 4, t = TRUE, flextable = FALSE, ft_args = list(), ...)
```

Arguments

<code>data</code>	A vector or data frame.
<code>digits</code>	Number of significant digits passed to summary.
<code>t</code>	If TRUE, variables are columns; otherwise variables are rows.
<code>flextable</code>	Return a flextable instead of a data frame.
<code>ft_args</code>	Named arguments for <code>flextable::flextable()</code> , excluding data.
<code>...</code>	Other arguments passed to summary.

Value

A data frame, including single-column inputs, or a flextable.

Examples

```
describe(cars)
describe(cars, t = FALSE)
describe(c(1, 2, NA))
if (requireNamespace("flextable", quietly = TRUE)) describe(cars, flextable = TRUE)
```

pin_get_or_create	<i>Retrieve or Create a Pin</i>
-------------------	---------------------------------

Description

This function attempts to retrieve a specified pin from a board. If the pin doesn't exist or the data is different, it creates a new pin.

Usage

```
pin_get_or_create(.data, .board, .name, type = "csv", ...)
```

Arguments

.data	The data to be pinned if a new pin is to be created.
.board	The board where the pin is located or should be created.
.name	The name of the pin to be retrieved or created.
type	The type of data being pinned, defaults to 'csv'.
...	Additional arguments to be passed to pins::pin_write.

Value

The version of the pin that was retrieved or created.

Examples

```
## Not run:
board <- pins::board_register_local()
data <- mtcars
pin_version <- pin_get_or_create(data, board, "mtcars_pin")

## End(Not run)
```

vars_to_date	<i>Construct calendar dates from columns</i>
--------------	--

Description

Parse daily, monthly, quarterly or annual observations without changing the source columns when drop_vars = FALSE. Month names and quarter ranges accept English and Spanish. Numeric, Q1/T1 and Roman quarter labels are supported, as are EViews labels such as 2020Q01 and 2020M02.

Usage

```
vars_to_date(
  tbl,
  year = NULL,
  quarter = NULL,
  month = NULL,
  day = NULL,
  date = NULL,
  drop_vars = TRUE,
  clean_names = FALSE,
  date_format = "%d-%m-%y",
  origin = "1900-01-01",
  .round = c("end", "middle", "start")
)
```

Arguments

<code>tbl</code>	A local data frame or tibble with unique column names.
<code>year, quarter, month, day</code>	One column name or position for each component. Day requires year and month; month or quarter requires year. When both month and quarter are supplied, they must agree.
<code>date</code>	Optional existing date column, as a name or position. Cannot be combined with component arguments. Accepts Date, POSIXt, numeric day counts, ISO dates, EViews periods, or character dates in <code>date_format</code> .
<code>drop_vars</code>	Drop the selected source columns, keeping the new date.
<code>clean_names</code>	Clean names with janitor before selecting columns.
<code>date_format</code>	Format for character dates, passed to <code>base::as.Date()</code> .
<code>origin</code>	Origin for numeric day counts. The historical default is 1900-01-01; use 1899-12-30 explicitly for modern Excel serial dates.
<code>.round</code>	Position within a month, quarter or year: end, middle or start. NULL selects start. Middle means the 15th of the month, the 15th of the middle month of a quarter, or July 1 for a year. Daily dates are not rounded.

Value

A copy of `tbl` with a Date column named `date` first. Invalid or missing selected components raise an error; dates never silently roll into a new month.

Examples

```
x <- data.frame(year = c(2020, 2021), quarter = c("Q1", "Enero-Marzo"))
vars_to_date(x, year = "year", quarter = "quarter", drop_vars = FALSE)
vars_to_date(data.frame(period = c("2020M02", "2021Q04")), date = "period")
```

Index

`%...>%`, [3](#)

`%<...%`, [2](#), [3](#)

`base::as.Date()`, [10](#)

`base::cut()`, [4](#)

`base::summary()`, [8](#)

`cut3`, [3](#)

`cut3_quantile`, [5](#)

`cut3_quantile()`, [4](#)

`db_connect`, [6](#)

`dbConnect`, [7](#)

`DBI::dbDisconnect()`, [6](#)

`describe`, [8](#)

`flextable::flextable()`, [8](#)

`pin_get_or_create`, [9](#)

`stats::quantile()`, [4](#), [5](#)

`vars_to_date`, [9](#)