Package 'datetoiso'

May 8, 2026

Type Package

Title Convert and Impute Dates to ISO 8601 Format and Reconcile Data Sets

Version 1.2.1

URL https://github.com/andzoluk

Language en-US

Description Provides tools for converting and imputing date values to the ISO 8601 standard format and for reconciling differences between two versions of a data set. The package automatically detects date patterns within data frame columns and converts them to consistent ISO-formatted dates, with optional imputation of missing day or month components based on user-defined rules. It also includes functionality to identify inserted, deleted, and updated records, as well as column- and value-level changes, when comparing old and new versions of a data frame. Only one date format may be applied within a single column.

License MIT + file LICENSE

Encoding UTF-8

Imports stringr, lubridate, data.table, dplyr, purrr, glue, magrittr

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

RoxygenNote 7.3.1

NeedsCompilation no

Author Lukasz Andrzejewski [aut, cre]

Maintainer Lukasz Andrzejewski <lukasz.coding@gmail.com>

Repository CRAN

Date/Publication 2025-12-16 15:10:02 UTC

Contents

check_day_correctly_entered_dmy

Check if Day Component is Valid in dmy date type

Description

This function checks whether the day component in a vector of date strings is valid, i.e., not exceeding the maximum number of days for the given month and year. It returns a logical vector indicating which elements have a correctly specified day.

Usage

```
check_day_correctly_entered_dmy(data_frame, column_name, separator = "-")
```

Arguments

data_frame	data frame
column_name	name of column that keeps dates to be imputed
separator	by default "-" it is a day-month-year separator, for example "2024-10-21" has "-" separator

Value

A logical vector

Author(s)

Lukasz Andrzejewski

check_day_correctly_entered_ymd

Check if Day Component is Valid in ymd date type

Description

This function checks whether the day component in a vector of date strings is valid, i.e., not exceeding the maximum number of days for the given month and year. It returns a logical vector indicating which elements have a correctly specified day.

Usage

```
check_day_correctly_entered_ymd(data_frame, column_name, separator = "-")
```

Arguments

data_frame	data frame
column_name	name of column that keeps dates to be imputed
separator	by default "-" it is a day-month-year separator, for example "2024-10-21" has "-" separator

Value

A logical vector

Author(s)

Lukasz Andrzejewski

---

check_if_month_year_entered

Check if a Vector Contains a Month and Year

Description

This function determines whether the elements of a vector contain a **month** and **year** in the specified order. It returns a logical vector indicating which elements meet this criterion.

Usage

```
check_if_month_year_entered(data_frame, column_name, separator = "-")
```

Arguments

data_frame	data frame
column_name	name of column that keeps dates to be imputed
separator	by default "-" it is a day-month-year separator, for example "2024-10-21" has "-" separator

Value

A logical vector

`check_if_only_year_entered` *Check if a Vector Contains only Year*

Description

This function determines whether the elements of a vector contain only **year** It returns a logical vector indicating which elements meet this criterion.

Usage

```
check_if_only_year_entered(
  data_frame,
  column_name,
  separator = "-",
  month = "UNK",
  day = "UN"
)
```

Arguments

`data_frame`	data frame
`column_name`	name of column that keeps dates to be imputed
`separator`	by default "-" it is a day-month-year separator, for example "2024-10-21" has "-" separator
`month`	by default "UNK" - the format of unknown month
`day`	by default "UN" - the format of unknown day

Value

A logical vector

Author(s)

Lukasz Andrzejewski

`check_if_year_month_day_entered` *Check if a vector contains a complete date*

Description

Check if a vector contains a complete date

Usage

```
check_if_year_month_day_entered(
  data_frame,
  column_name,
  separator = "-",
  date_format = "ymd",
  year = "UNKN",
  month = "UNK",
  day = "UN"
)
```

Arguments

`data_frame`	data frame
`column_name`	name of column that keeps dates to be imputed
`separator`	by default "-" it is a day-month-year separator, for example "2024-10-21" has "-" separator
`date_format`	by default "ymd". choose between ymd (if first year, then month then day) and dmy (if first day, then month then year)
`year`	by default "UNKN" - the format of unknown year
`month`	by default "UNK" - the format of unknown month
`day`	by default "UN" - the format of unknown day

Value

A logical vector

Author(s)

Lukasz Andrzejewski

check_if_year_month_entered *Check if a Vector Contains a Year and Month*

Description

This function determines whether the elements of a vector contain a **year** and **month** in the specified order. It returns a logical vector indicating which elements meet this criterion.

Usage

```
check_if_year_month_entered(data_frame, column_name, separator = "-")
```

Arguments

data_frame	data frame
column_name	name of column that keeps dates to be imputed
separator	by default "-" it is a day-month-year separator, for example "2024-10-21" has "-" separator

Value

A logical vector

Author(s)

Lukasz Andrzejewski

choose_dmy_format *Get TRUE if date format is dmy*

Description

Get TRUE if date format is dmy

Usage

```
choose_dmy_format(df_column)
```

Arguments

df_column	data frame date column or vector with dates

Value

logical vector, TRUE if most probable date format is DMY

Author(s)

Lukasz Andrzejewski

`choose_dym_format` *Get TRUE if date format is dym*

Description

Get TRUE if date format is dym

Usage

```
choose_dym_format(df_column)
```

Arguments

`df_column`	data frame date column or vector with dates

Value

logical vector, TRUE if most probable date format is DYM

Author(s)

Lukasz Andrzejewski

`choose_mdy_format` *Get TRUE if date format is mdy*

Description

Get TRUE if date format is mdy

Usage

```
choose_mdy_format(df_column)
```

Arguments

`df_column`	data frame date column or vector with dates

Value

logical vector, TRUE if most probable date format is MDY

Author(s)

Lukasz Andrzejewski

choose_myd_format *Get TRUE if date format is myd*

Description

Get TRUE if date format is myd

Usage

```
choose_myd_format(df_column)
```

Arguments

df_column	data frame date column or vector with dates

Value

logical vector, TRUE if most probable date format is MYD

Author(s)

Lukasz Andrzejewski

choose_ydm_format *Get TRUE if date format is ydm*

Description

Get TRUE if date format is ydm

Usage

```
choose_ydm_format(df_column)
```

Arguments

df_column	data frame date column or vector with dates

Value

logical vector, TRUE if most probable date format is YDM

Author(s)

Lukasz Andrzejewski

choose_ymd_format *Get TRUE if date format is ymd*

Description

Get TRUE if date format is ymd

Usage

```
choose_ymd_format(df_column)
```

Arguments

df_column	data frame date column or vector with dates

Value

logical vector, TRUE if most probable date format is YMD

Author(s)

Lukasz Andrzejewski

clean_date *Prepare and normalize date-like strings before YMD conversion*

Description

This function applies a series of cleaning and normalization steps to strings representing dates. It is intended for use before parsing dates into a YMD (year–month–day) format. The function standardizes month names, trims whitespace, removes invalid characters, and handles strings that contain a letter "T" (common in timestamp formats).

Usage

```
clean_date(df_column)
```

Arguments

df_column	A character vector or data frame column containing raw date-like strings to be cleaned.

Details

The processing includes:

- Converting full month names to abbreviated forms (via `get_abbreviated_month_name()`).
- Limiting the string to the first 12 characters (via `get_up_to_12_char()`).
- Removing non-date characters (via `remove_no_date_characters()`).
- Trimming whitespace at the start and end of the string.
- Handling timestamps or strings containing the letter "T":
  - If "T" appears exactly once and the string does not contain "August" or "October", keep only the substring before "T".
  - If "T" appears multiple times, remove the unnecessary trailing part using `remove_unnecessary_part_of_date()`.
- If the first token of the string (separated by a space) is longer than four characters, return only that first token.

Value

A character vector of cleaned date strings, with a maximum length of 12 characters, trimmed of whitespace, and with any timestamp-like "T" components removed when appropriate.

Author(s)

Lukasz Andrzejewski

Examples

```
clean_date(c("2024-01-10T15:30:00", "2024 AUGUST 12", "20250101"))
```

`compare_rows_with_same_index` *Identifies and summarizes value-level changes between two datasets*

Description

This function compares corresponding rows of two data frames and generates two columns:

Usage

```
compare_rows_with_same_index(new_df, old_df)
```

Arguments

`new_df`	A data frame containing the most recent version of the data.
`old_df`	A data frame containing the preceding version of the data, used as the reference for comparison.

Details

* **changed_cols** — a list-column containing the names of variables in which at least one change has been detected. * **change_details** — a list-column describing the specific modifications for each changed variable, expressed as "previous value - new value".

Value

A data frame augmented with two additional columns: ‘changed_cols‘ and ‘change_details‘

Author(s)

Lukasz Andrzejewski

`dfiso` *Function recognize date variables and modify them to ISO standard ("International Organization for Standardization")*

Description

Function recognize date variables and modify them to ISO standard ("International Organization for Standardization")

Usage

```
dfiso(df)
```

Arguments

`df`	data frame or variable/s, for example data.frame(date=c("12-Mar-2021","01-Jan-2023"))

Value

dates formatted to ISO standard (yyyy-mm-dd)

Author(s)

Lukasz Andrzejewski

Examples

```
# data frame with different formatted dates
dfiso(data.frame(date1=c("13-02-2022","13/Feb/2022","13-Feb-2022")))
```

find_dmy_date_format *Find DMY dates only*

Description

Find DMY dates only

Usage

```
find_dmy_date_format(df_column)
```

Arguments

df_column	data frame date column or vector with dates

Value

logical vector, TRUE if date format is DMY

Author(s)

Lukasz Andrzejewski

find_dym_date_format *Find DYM dates only*

Description

Find DYM dates only

Usage

```
find_dym_date_format(df_column)
```

Arguments

df_column	data frame date column or vector with dates

Value

logical vector, TRUE if date format is DYM

Author(s)

Lukasz Andrzejewski

find_mdy_date_format *Find MDY dates only*

Description

Find MDY dates only

Usage

```
find_mdy_date_format(df_column)
```

Arguments

df_column	data frame date column or vector with dates

Value

logical vector, TRUE if date format is MDY

Author(s)

Lukasz Andrzejewski

find_myd_date_format *Find MYD dates only*

Description

Find MYD dates only

Usage

```
find_myd_date_format(df_column)
```

Arguments

df_column	data frame date column or vector with dates

Value

logical vector, TRUE if date format is MYD

Author(s)

Lukasz Andrzejewski

find_only_dates *Return TRUE if data frame column or vector contains date*

Description

Return TRUE if data frame column or vector contains date

Usage

```
find_only_dates(df_column)
```

Arguments

df_column	data frame date column or vector with dates

Value

logical vector, return TRUE if number of characters is higher than 5, contains digits and special characters or month names

Author(s)

Lukasz Andrzejewski

find_unknow_date *Find Unknown date, defined as UN or UNK*

Description

Find Unknown date, defined as UN or UNK

Usage

```
find_unknow_date(df_column)
```

Arguments

df_column	data frame date column or vector with dates

Value

logical vector, TRUE if "un" character is found but not "jun"

Author(s)

Lukasz Andrzejewski

`find_ydm_date_format` *Find YDM dates only*

Description

Find YDM dates only

Usage

```
find_ydm_date_format(df_column)
```

Arguments

`df_column`	data frame date column or vector with dates

Value

logical vector, TRUE if date format is YDM

Author(s)

Lukasz Andrzejewski

`find_ymd_date_format` *Find YMD dates only*

Description

Find YMD dates only

Usage

```
find_ymd_date_format(df_column)
```

Arguments

`df_column`	data frame date column or vector with dates

Value

logical vector, TRUE if date format is YMD

Author(s)

Lukasz Andrzejewski

`get_abbreviated_month_name` *Replace full month name by abbreviated month name*

Description

Replace full month name by abbreviated month name

Usage

```
get_abbreviated_month_name(df_column)
```

Arguments

`df_column`	data frame date column or vector with dates

Value

vector, if any full length month name, then replace by abbreviated month name

Author(s)

Lukasz Andrzejewski

`get_full_name_months_sep_by_vertical_bar` *Get vector with full name of months separated by vertical bar*

Description

Get vector with full name of months separated by vertical bar

Usage

```
get_full_name_months_sep_by_vertical_bar()
```

Value

full names and abbreviations of months separated by vertical bar

Author(s)

Lukasz Andrzejewski

`get_max_score_within_data_formats` *Score each of date format ymd, ydm, dmy, dym, mdy, myd and return only the highest score*

Description

Score each of date format ymd, ydm, dmy, dym, mdy, myd and return only the highest score

Usage

```
get_max_score_within_data_formats(df_column)
```

Arguments

`df_column`	data frame date column or vector with dates

Value

return score of most probable date format

Author(s)

Lukasz Andrzejewski

`get_months` *List month names: full names and abbreviated names in lower case*

Description

List month names: full names and abbreviated names in lower case

Usage

```
get_months()
```

Value

full names and abbreviations of months

Author(s)

Lukasz Andrzejewski

get_months_full_names *List month names: full names in lower case*

Description

List month names: full names in lower case

Usage

```
get_months_full_names()
```

Value

full names of months

Author(s)

Lukasz Andrzejewski

get_months_sep_by_vertical_bar *Get vector with full and abbreviated name of months separated by vertical bar*

Description

Get vector with full and abbreviated name of months separated by vertical bar

Usage

```
get_months_sep_by_vertical_bar()
```

Value

full names and abbreviations of months separated by vertical bar

Author(s)

Lukasz Andrzejewski

get_number_of_symbols_in_string *Function to find number of symbols in date*

Description

Function to find number of symbols in date

Usage

```
get_number_of_symbols_in_string(df_column, symbol = "T")
```

Arguments

df_column	data frame date column or vector with dates
symbol	symbol that needs to be found, by default "T"

Value

number of found symbols

Author(s)

Lukasz Andrzejewski

get_same_class *Harmonizes variable classes between two data frames*

Description

This function aligns the classes of variables in ‘old_df‘ with those of their corresponding variables in ‘new_df‘. For each column name shared across both data frames, the function detects the class in ‘new_df‘ and coerces the matching column in ‘old_df‘ to the same class. Supported conversions include ‘character‘, ‘numeric‘, ‘integer‘, ‘logical‘, ‘factor‘, and ‘Date‘. Any variable whose class is not explicitly handled is left unchanged.

Usage

```
get_same_class(old_df, new_df)
```

Arguments

old_df	A data frame containing the preceding version of the data, used as the reference for comparison.
new_df	A data frame containing the most recent version of the data.

Value

A modified version of ‘old_df‘ in which all shared columns are coerced to match the variable classes of ‘new_df‘.

Author(s)

Lukasz Andrzejewski

---

get_up_to_12_char *function return observations with up to 12 characters*

Description

function return observations with up to 12 characters

Usage

```
get_up_to_12_char(df_column)
```

Arguments

df_column	data frame column or vector to extract observarions up to 12 characters

Value

return up to 12 characters

Author(s)

Lukasz Andrzejewski

---

has_dash_or_slash_or_white_space_characters_or_months_separated_by_vertical_bar *Function return special characters and months separated by vertical bars*

Description

Function return special characters and months separated by vertical bars

Usage

```
has_dash_or_slash_or_white_space_characters_or_months_separated_by_vertical_bar(

)
```

Value

special characters and months: "-|\/|\w+\s+|january|february|march|april|may|june|july|august|september|october|november|de

Author(s)

Lukasz Andrzejewski

---

has_dash_or_slash_or_white_space_characters_separated_by_vertical_bar *Function return special characters separated by vertical bars*

Description

Function return special characters separated by vertical bars

Usage

```
has_dash_or_slash_or_white_space_characters_separated_by_vertical_bar(
  special_characters = c("-", "\\/", "\\w+\\s+")
)
```

Arguments

special_characters	by default dash, slash, white space characters

Value

special characters: "-|\/|\w+\s+"

Author(s)

Lukasz Andrzejewski

---

impute_date *Impute Missing Components in Partial Date Strings*

Description

This function imputes missing **month** and/or **day** components in partial date strings where the **year** is known. It assumes input dates are provided in either the *dmy* format (day-month-year) **or** the *ymd* format (year-month-day) and does not process datetime values or strings containing time components or non-date characters.

Usage

```
impute_date(
  data_frame,
  column_name,
  date_format = "ymd",
  separator = "-",
  year = "UNKN",
  month = "UNK",
  day = "UN",
  min_max = "min",
  suffix = "_DT"
)
```

Arguments

data_frame	data frame
column_name	name of column that keeps dates to be imputed
date_format	by default "ymd". choose between ymd (if first year, then month then day) and dmy (if first day, then month then year)
separator	by default "-" it is a day-month-year separator, for example "2024-10-21" has "-" separator
year	by default "UNKN" - the format of unknown year
month	by default "UNK" - the format of unknown month
day	by default "UN" - the format of unknown day
min_max	by default "min". controlling imputation direction."min" - Impute the earliest possible date "max"‘ - Impute the latest possible date
suffix	by default "_DT" - new imputed date is named as source variable with suffix

Details

If the **year** is missing or explicitly marked as unknown (e.g., ‘"UNKN"‘), the function returns ‘NA‘. When the **month** is missing, the function imputes **January (01)** as the default month. When the **day** is missing, it imputes the **first day of the month (01)**.

Any datetime strings (e.g., ‘"NA-01-2025T11:10:00"‘) must be preprocessed to remove the time component before applying this function (e.g., convert to ‘"NA-01-2025"‘).

In addition to imputing the date, the function creates an accompanying **flag variable** named as: ‘"<source_variable>_<suffix>F"‘. This flag variable indicates the type of imputation performed:

- ‘NA‘ — No imputation was performed (the original date was complete).
- ‘"D"‘ — The **day** component was imputed.
- ‘"M"‘ — The **month** component were imputed.
- ‘"D, M"‘ — Both **month** and **day** components were imputed.

Value

A data frame identical to the input, with an additional column representing the imputed values. The imputed column name is constructed by appending the suffix "_imputed" to the source variable name.

Author(s)

Lukasz Andrzejewski

Examples

```
impute_date(data_frame = data.frame(K = c('2025 11 UN', '2025 UNK 23')),
column_name = "K", separator = " ")
```

`impute_date_dmy` *Impute Missing Components in Partial Date Strings*

Description

This function imputes missing **month** and/or **day** components in partial date strings where the **year** is known. It assumes input dates are provided in the *dmy* format (day-month-year) and does not process datetime values or strings containing time components or non-date characters.

Usage

```
impute_date_dmy(
  data_frame,
  column_name,
  separator = "-",
  year = "UNKN",
  month = "UNK",
  day = "UN",
  min_max = "min",
  suffix = "_DT"
)
```

Arguments

`data_frame`	data frame
`column_name`	name of column that keeps dates to be imputed
`separator`	by default "-" it is a day-month-year separator, for example "2024-10-21" has "-" separator
`year`	by default "UNKN" - the format of unknown year
`month`	by default "UNK" - the format of unknown month
`day`	by default "UN" - the format of unknown day
`min_max`	by default "min". controlling imputation direction."min" - Impute the earliest possible date "max"‘ - Impute the latest possible date
`suffix`	by default "_DT" - new imputed date is named as source variable with suffix

Details

If the **year** is missing or explicitly marked as unknown (e.g., ‘"UNKN"‘), the function returns ‘NA‘. When the **month** is missing, the function imputes **January (01)** as the default month. When the **day** is missing, it imputes the **first day of the month (01)**.

Any datetime strings (e.g., ‘"NA-01-2025T11:10:00"‘) must be preprocessed to remove the time component before applying this function (e.g., convert to ‘"NA-01-2025"‘).

In addition to imputing the date, the function creates an accompanying **flag variable** named as: ‘"<source_variable>_<suffix>F"‘. This flag variable indicates the type of imputation performed:

- ‘NA‘ — No imputation was performed (the original date was complete or missing year).
- ‘"D"‘ — The **day** component was imputed.
- ‘"M"‘ — The **month** component was imputed.
- ‘"D, M"‘ — Both **month** and **day** components were imputed.

Value

A data frame identical to the input, with an additional column representing the imputed values. The imputed column name is constructed by appending the suffix "_imputed" to the source variable name.

Author(s)

Lukasz Andrzejewski

`impute_date_ymd` *Impute Missing Components in Partial Date Strings*

Description

This function imputes missing **month** and/or **day** components in partial date strings where the **year** is known. It assumes input dates are provided in the *ymd* format (year-month-day) and does not process datetime values or strings containing time components or non-date characters.

Usage

```
impute_date_ymd(
  data_frame,
  column_name,
  separator = "-",
  year = "UNKN",
  month = "UNK",
  day = "UN",
  min_max = "min",
  suffix = "_DT"
)
```

Arguments

data_frame	data frame
column_name	name of column that keeps dates to be imputed
separator	by default "-" it is a day-month-year separator, for example "2024-10-21" has "-" separator
year	by default "UNKN" - the format of unknown year
month	by default "UNK" - the format of unknown month
day	by default "UN" - the format of unknown day
min_max	by default "min". controlling imputation direction."min" - Impute the earliest possible date "max"‘ - Impute the latest possible date
suffix	by default "_DT" - new imputed date is named as source variable with suffix

Details

If the **year** is missing or explicitly marked as unknown (e.g., ‘"UNKN"‘), the function returns ‘NA‘. When the **month** is missing, the function imputes **January (01)** as the default month. When the **day** is missing, it imputes the **first day of the month (01)**.

Any datetime strings (e.g., ‘"2025-01-NAT11:10:00"‘) must be preprocessed to remove the time component before applying this function (e.g., convert to ‘"2025-01-NA"‘).

In addition to imputing the date, the function creates an accompanying **flag variable** named as: ‘"<source_variable>_<suffix>F"‘. This flag variable indicates the type of imputation performed:

- ‘NA‘ — No imputation was performed (the original date was complete or missing year).
- ‘"D"‘ — The **day** component was imputed. The **month** component was imputed.
- ‘"M"‘ — The **month** component were imputed.
- ‘"D, M"‘ — Both **month** and **day** components were imputed.

Value

A data frame identical to the input, with an additional column representing the imputed values. The imputed column name is constructed by appending the suffix "_imputed" to the source variable name.

Author(s)

Lukasz Andrzejewski

`reconcile` *Reconcile Two Versions of a Data Frame*

Description

This function compares a new and an old version of a data set to identify inserted, deleted, and updated records, as well as column-level changes. The comparison can be performed using a specified index column (or columns), or—if no index is provided—based on a full-row comparison across all common columns.

Usage

```
reconcile(new_df, old_df, index = NA, lookup_columns = NA)
```

Arguments

`new_df`	A data frame containing the most recent version of the data.
`old_df`	A data frame containing the preceding version of the data, used as the reference for comparison.
`index`	A character vector specifying the variable(s) that uniquely identify records (e.g., ‘"recordid"‘). If ‘NA‘, all common columns are used as the matching key, but some enhanced functionality (such as detecting newly added or removed rows) will not be available.
`lookup_columns`	A character vector specifying which columns should be compared. By default ‘NA‘, meaning that all columns common to both ‘new_df‘ and ‘old_df‘ are used. If specific column names are provided, comparisons are restricted to those columns.

Details

When ‘index‘ is supplied, rows are matched by the specified index variable(s), allowing the function to detect newly added records, removed records, and detailed field-level changes. When ‘index = NA‘, the function falls back to a full reconciliation based on the auxiliary comparison routine, using all common columns as the key.

Column comparison is further controlled by ‘lookup_columns‘: if this argument is left as ‘NA‘, all columns common to ‘new_df‘ and ‘old_df‘ are evaluated; otherwise, only the specified subset of columns is compared.

Value

A data frame summarizing the reconciliation results. For each record, the output includes the current values, index variables, detected status (‘"NEW"‘, ‘"DELETED"‘, ‘"UPDATED"‘, ‘"UNCHANGED"‘), the set of changed columns, and a human-readable description of the differences.

Author(s)

Lukasz Andrzejewski

Examples

```
reconcile(data.frame(col1 = c("AA", "B"), id = c(1, 2)),
data.frame(col1 = c("A", "B"), id = c(1, 3)), index = "id")
```

`reconcile_without_index` *Reconcile Two Data Frames*

Description

This function compares two data frames —'new_df' (the updated version) and 'old_df' (the previous version) —to identify differences between them. The comparison can be performed across all shared columns or restricted to a specified subset of columns.

Usage

```
reconcile_without_index(old_df, new_df, lookup_columns = NA)
```

Arguments

`old_df`	A data frame containing the preceding version of the data, used as the reference for comparison.
`new_df`	A data frame containing the most recent version of the data.
`lookup_columns`	A character vector specifying which columns should be used for comparison. By default 'NA', meaning that all columns common to both 'new_df' and 'old_df' are included. If one or more column names are provided, only those columns will be compared.

Value

A data frame summarizing differences between 'new_df' and 'old_df', including which columns changed and the details of those changes.

Author(s)

Lukasz Andrzejewski

`remove_no_date_characters` *Remove unnecessary characters from date-like strings*

Description

This function cleans a character vector or data frame column containing date-like strings by removing all characters that are not needed for parsing or recognizing dates. It preserves:

- Digits (0–9)
- Letters that appear in any full month name (e.g., "January" → "J, A, N, U, R, Y")
- Selected extra allowed characters: space (" "), dash ("-"), slash ("/"), and "k"/"K"

All other characters (symbols, punctuation, letters not in month names) are removed.

Usage

```
remove_no_date_characters(df_column)
```

Arguments

`df_column`	A character vector (or data frame column) containing date-like strings. Factors will be coerced to character. NA values are preserved.

Details

The function works as follows:

1. Converts input to character vector.
2. Generates the set of letters present in all English month names (case-insensitive).
3. Constructs a regex pattern to match all characters that are NOT digits, allowed letters, or allowed extra symbols.
4. Uses `stringr::str_replace_all()` to remove unwanted characters.

Value

A character vector of the same length as `df_column`, with unwanted characters removed. Only digits, letters from month names, and selected extra characters are kept.

Author(s)

Lukasz Andrzejewski

remove_unnecessary_part_of_date

Get substring of date to eliminate unnecessary part

Description

Get substring of date to eliminate unnecessary part

Usage

```
remove_unnecessary_part_of_date(df_column, symbol = "T")
```

Arguments

df_column	date column or vector with dates
symbol	symbol that needs to be found, by default "T"

Value

substring of date from position 1 to position where last "symbol" is located

Author(s)

Lukasz Andrzejewski

viso

transform date vector to date vector in ISO standard ("International Organization for Standardization")

Description

transform date vector to date vector in ISO standard ("International Organization for Standardization")

Usage

```
viso(df_column)
```

Arguments

df_column	vector or string

Value

dates formatted to ISO standard (yyyy-mm-dd)

Author(s)

Lukasz Andrzejewski

Examples

```
#day month year vector
viso(c("12Mar2022","21Feb2022"))

#day month year vector in different formats
viso(c("12Mar2022","21-02-2022"))

#month year day vector
viso(c("Mar-2022-12","Feb-2022-21"))
```

Index