Package: dmhct (via r-universe)

October 14, 2024

Title A Data Model Package for the MLinHCT Project

Version 1.0.0

Description Extracts, Loads, and Transforms data from the SQL Server containing HCT data to a `dm` object with cleaned tables.

License AGPL (>= 3)

URL https://github.com/jesse-smith/dmhct, https://jesse-smith.github.io/dmhct/

BugReports https://github.com/jesse-smith/dmhct/issues

Depends R (>= 4.0)

Imports bit64, checkmate, data.table, dbplyr, dm (>= 1.0.0), dplyr, fs, janitor, lifecycle, lubridate, methods, odbc, purrr, R6, rlang, stringi, stringr, vctrs

Suggests cli, devtools, DiagrammeR, DiagrammeRsvg, forcats, lme4, qs, renv, stats, tidyr, usethis

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Repository https://jesse-smith.r-universe.dev

RemoteUrl https://github.com/jesse-smith/dmhct

RemoteRef HEAD

RemoteSha e024cd902179c4107db120020cd0d2ba1fb17220

Contents

`con_irb_mlinhct` *Connect to SQL Server Where IRB_MLinHCT Data is Stored*

Description

Connect to SQL Server Where IRB_MLinHCT Data is Stored

Usage

```
con_irb_mlinhct(
  server = "SVWPBMTCTDB01",
  database = "IRB_MLinHCT",
  trusted_connection = TRUE,
  dsn = NULL,
  ...
)
```

Arguments

`server`	`[chr(1)]` Name of server
`database`	`[chr(1)]` Name of database
`trusted_connection`	`[lgl(1)]` Whether this is a "trusted connection"; using Windows Authentication means it is such a connection.
`dsn`	`[chr(1)]` A DSN name to use for connection; if provided, the above arguments are ignored.
`...`	Additional named arguments to pass to `odbc::dbConnect()`

Value

`[Microsoft SQL Server]` An ODBC connection object

`con_sql_server` *Connect to SQL Server Where Data is Stored*

Description

Connect to SQL Server Where Data is Stored

Usage

```
con_sql_server(dbname = c("IRB_MLinHCT", "EDW"))
```

Arguments

`dbname`	`[chr(1)]` The name of the database to connect to

Value

`[Microsoft SQL Server]` An ODBC connection object

`con_stjude_edw` *Connect to SQL Server Where EDW Data is Stored*

Description

Connect to SQL Server Where EDW Data is Stored

Usage

```
con_stjude_edw(
  server = "stjude-edw.database.windows.net",
  database = "EDW",
  authentication = "ActiveDirectoryIntegrated",
  ...
)
```

Arguments

`server`	`[chr(1)]` Name of server
`database`	`[chr(1)]` Name of database
`authentication`	`[chr(1)]` The authentication type to use; default is `ActiveDirectoryIntegrated`
`...`	Additional named arguments to pass to `odbc::dbConnect()`

Value

`[Microsoft SQL Server]` An ODBC connection object

dm_collect *Collect All Tables in a* dm *Object*

Description

Collect All Tables in a dm Object

Usage

```
dm_collect(dm_remote, data_table = FALSE)
```

Arguments

dm_remote	[dm] A dm object connected to a remote source
data_table	[lgl] Whether to return a data.table. If FALSE, (the default), will return a tibble instead.

Value

[dm] A new dm object containing the collected (local) tables

dm_combine *Combine Table with Like Information*

Description

Combine Table with Like Information

Usage

```
dm_combine(dm_std = dm_standardize(), quiet = FALSE)
```

Arguments

dm_std	A standardized dm object. Standardization is necessary to ensure columns are all of the same type.
quiet	Should update messages be suppressed?

Value

The updated dm object

`dm_compute` *Compute All Tables in a* `dm` *Object*

Description

Compute All Tables in a `dm` Object

Usage

```
dm_compute(dm_remote, quiet = TRUE)
```

Arguments

`dm_remote`	`[dm]` A `dm` object connected to a remote source
`quiet`	`[lgl(1)]` Should messages be suppressed during computation?

Value

`[dm]` The updated object with tables computed

`dm_disconnect` *Disconnect a* `dm` *Object from the Remote Server*

Description

Disconnect a `dm` Object from the Remote Server

Usage

```
dm_disconnect(dm)
```

Arguments

`dm`	`[dm]` The `dm` object to disconnect

Value

`[dm]` The `dm` (invisibly)

dm_elt — *Extract, Load, and Transform Remote Tables to Local Source*

Description

`dm_elt()` encompasses the entire legacy dmhct pipeline; however, this pipeline is deprecated no longer under active development. While this function will be retained for backwards compatibility, it is strongly recommended that new code use the new pipeline instead.

Usage

```
dm_elt(dm_remote = dm_sql_server(), reset = FALSE, close = NULL)
```

Arguments

`dm_remote`	`[dm]` Remote `dm` object containing HCT data
`reset`	`[lgl(1)]` Should the cache be reset to the current results, even if inputs have not changed? This is useful if data processing logic has changed, but the underlying data have not.
`close`	`[lgl(1)]` Whether to close the SQL Server connection on exit. `NULL` closes if `dm_remote` has attribute `default == TRUE` and leaves open otherwise.

Value

`[dm]` A `dm` object

dm_extract — *Extract Remote Tables from SQL Server for MLinHCT*

Description

`dm_extract()` extracts and (optionally) loads the remote database housing the MLinHCT into the current R session. Unless `.legacy = TRUE`, column and table names are standardized during extraction, but no other operations are performed. When `.legacy = TRUE`, the legacy version of `dm_extract()` is used; see details for this behavior. Note that legacy behavior will be deprecated and eventually removed in future releases, so it is strongly recommended that any new code use `.legacy = FALSE`.

Usage

```
dm_extract(
  dm_remote = dm_sql_server(),
  ...,
  .collect = TRUE,
  .legacy = FALSE,
```

```
  .reset = FALSE,
  .excl_dsmb = FALSE,
  .quiet = FALSE,
  reset = .reset
)
```

Arguments

dm_remote	[dm] A dm object connected to the remote SQL server database
...	Names of tables to select; if provided, only these tables will be extracted
.collect	[lgl] Indicates whether the extracted data should be loaded onto the local machine (TRUE by default)
.legacy	[lgl] Should the legacy version of dm_extract() be used? Will be deprecated in a future release, along with .reset.
.reset	[lgl] Should the legacy cache be forced to reset? Only applicable if .legacy = TRUE; ignored otherwise. Will be deprecated in a future release, along with .legacy.
.excl_dsmb	[lgl] [**Deprecated**] This information is no longer available in the remote database.
.quiet	Should status messages be suppressed?
reset	[lgl] [**Deprecated**] Please use .reset instead. Current behavior will only consider this argument if .reset is unchanged from the default.

Details

Legacy behavior is more opinionated than the current version of dm_extract(). First, only a subset of tables and columns are extracted. Second, HLA tables and Cerner tables are combined into a single HLA table and a single Cerner table. Third, some column standardization occurs (though it is limited to simple as() transformations, trimws(toupper(x)) on character variables, and replacement of implicit missing values with explicit NAs.) Finally, some filtering of "uninformative" observations may occur. In the current pipeline, these changes are deferred to later steps to give more control to the user.

Value

[dm] A dm object with all tables and columns extracted from the remote source.

---

dm_extract_legacy *Select and Convert Table + Columns From Remote Source*

Description

dm_extract_legacy() is a previous, less extensible version of dm_extract(). It selects tables and columns of potential interest. It combines all Cerner tables into one and joins HLA tables.

Usage

```
dm_extract_legacy(dm_remote = dm_sql_server(), collect = TRUE, reset = FALSE)
```

Arguments

dm_remote	[dm] A dm object connected to the SQL Server for MLinHCT
collect	[lgl(1)] Should tables be collected locally on output?
reset	[lgl(1)] Should the cache be reset to the current results, even if inputs have not changed? This is useful if data processing logic has changed, but the underlying data have not.

Value

[dm] The updated dm object

---

dm_hct — *Extract, Standardize, and Combine Tables from the MLinHCT Database*

Description

dm_hct() chains together dm_extract(), dm_standardize(), and dm_combine() to provide a single wrapper function for data preparation.

Usage

```
dm_hct(dm_remote = dm_sql_server(), ..., .excl_dsmb = FALSE, .quiet = FALSE)
```

Arguments

dm_remote	[dm] A dm object connected to the remote SQL server database
...	Names of tables to select; if provided, only these tables will be extracted
.excl_dsmb	[lgl] **[Deprecated]** This information is no longer available in the remote database.
.quiet	Should status messages be suppressed?

Value

The prepared dm object

dm_is_remote — *Check Whether a* dm *Object is Connected to a Remote Server*

Description

Check Whether a dm Object is Connected to a Remote Server

Usage

```
dm_is_remote(dm)
```

Arguments

dm	The dm object to check

Value

[lgl(1)] Whether the dm is remote or not

dm_pivot — *Pivot Tables in Entity-Attribute-Value Format*

Description

Pivot Tables in Entity-Attribute-Value Format

Usage

```
dm_pivot(dm_cmb = dm_combine(), quiet = FALSE)
```

Arguments

dm_cmb	A dm object with combined tables. This is necessary b/c the pivoted tables are created by dm_combine().
quiet	Should update messages be suppressed?

Value

The updated dm object

dm_sql_server *Create a* dm *Object of HCT Data from Connection to SQL Server*

Description

Create a dm Object of HCT Data from Connection to SQL Server

Usage

```
dm_sql_server(con = con_sql_server(), quiet = FALSE)
```

Arguments

con	[Microsoft SQL Server] An ODBC connection to a SQL Server database
quiet	Should update messages be suppressed?

Value

[dm] A dm containing HCT data

dm_standardize *Standardize Column Values in a local* dm *for MLinHCT*

Description

dm_standardize() takes a local version of the SQL server as input and standardizes all columns across all tables. Standardization procedures are based on both column type and the typing prefix of the column name. Specifically, columns are standardized using the following workflow:

1. Columns with type character or chr/cat/lgl/mcat/intvl prefixes are passed to std_chr()
2. Columns with type logical or the lgl prefix are passed to std_lgl()
3. Columns with type numeric or integer, or num/pct prefixes, are passed to std_num()
4. Columns with the intvl prefix are passed to std_intvl()
5. Column with types Date, POSIXct, or POSIXlt, or with dt/dttm/date prefixes, are passed to std_date()

After standardization, tables are sorted into alphabetical order before returning.

Usage

```
dm_standardize(dm_local = dm_extract(), quiet = FALSE)
```

Arguments

`dm_local`	A local `dm` object containing MLinHCT data from `dm_extract()`
`quiet`	Whether to suppress progress messages

Value

The input `dm` with standarized column values

`dm_transform` *Transform Tables to Analysis-Friendly Format*

Description

A previous version of the dmhct pipeline performed all transformations of tables simultaneously; to ensure backwards compatibility, this behavior has been retained in `dm_transform()`. However, it is strongly recommended that new code not use `dm_transform()` and instead use the updated pipeline.

Usage

```
dm_transform(dm_local = dm_extract_legacy(), reset = FALSE)
```

Arguments

`dm_local`	`[dm]` A local `dm` object from `dm_extract()`
`reset`	`[lgl(1)]` Should the cache be reset to the current results, even if inputs have not changed? This is useful if data processing logic has changed, but the underlying data have not.

Value

`[dm]` The updated `dm` object

`intvl_to_matrix` *Convert Standardized Intervals to Matrix Format*

Description

`intvl_to_matrix()` converts interval representation standardized by `std_invl()` to a 4-column `numeric` matrix. Columns represent open or closed bounds and the location of those bounds.

Usage

```
intvl_to_matrix(x)
```

Arguments

x	A character vector of standardized intervals

Value

A 4-column numeric matrix:

- left_closed: Whether the left bound is closed or open
- left_bound: The left bound of the interval
- right_bound: The right bound of the interval
- right_closed: Whether the right bound is closed or open

---

na_patterns *Common Patterns Representing Missing Data*

Description

na_patterns is a collection of regular expression that commonly represent missing data, especially when the character vector should be converted to something else. These are designed to match strings that have already been standardized.

Usage

```
na_patterns
```

Format

An object of class character of length 8.

---

non_numeric *Extract Values That Cannot Be Converted to* numeric

---

Description

non_numeric() is designed primarily for interactive checking of numeric conversions. It helps quickly determine what values in a vector cannot be converted to numeric (either directly or via std_num()); this is particularly useful for checking steps of a data cleaning pipeline.

Usage

```
non_numeric(x, unique = TRUE, sort = unique, std_num = FALSE)
```

Arguments

`x`	A vector
`unique`	Whether unique values should be returned; if `FALSE`, all values are returned
`sort`	Whether return values should be sorted; most useful when `unique = TRUE`
`std_num`	Whether to use `std_num()` for numeric conversion; if `FALSE`, conversion is performed directly by `as.numeric()` (with warnings suppressed)

Value

The values of `x` that resulted in `NA_real_` after conversion; this includes any `NA` values in `x` before conversion

---

`std_chr` *Standardize* `character` *Vectors*

Description

`std_chr()` standardizes `character` vectors to ASCII text with no unnecessary whitespace and a given case. By default, it will retain newlines inside text, though it will condense consecutive newlines and any carriage returns into a single newline.

Usage

```
std_chr(
  x,
  case = c("upper", "lower", "title", "sentence"),
  keep_inner_newlines = TRUE,
  na = "^$"
)
```

Arguments

`x`	A character vector
`case`	The case to convert to. `NULL` will skip case conversion.
`keep_inner_newlines`	Whether to retain line breaks inside text. `FALSE` will treat newlines and carriage returns identically to any other whitespace.
`na`	Regex patterns to consider `NA`. Passed to `stringr::str_detect()`. Can be a vector of patterns.

Value

The standardized character vector

`std_date` *Parse Dates to Standard Format*

Description

`std_date` standardizes a date vector and returns a vector in `Date` or `POSIXct` format, depending on whether there is sub-daily information available in the data.

Usage

```
std_date(
  x,
  force = c("none", "dt", "dttm"),
  orders = c("mdy", "dmy", "ymd", "mdyr", "dmyr", "ymdr", "mdyR", "dmyR", "ymdR", "mdyT",
    "dmyT", "ymdT", "mdyTz", "dmyTz", "ymdTz", "Tmdyz", "Tdmyz", "Tymdz", "mdyRz",
    "dmyRz", "ymdRz", "mdyrz", "dmyrz", "ymdrz", "Tmdy", "Tdmy", "Tymd", "Tmdyz",
    "Tdmyz", "Tymdz"),
  tz_heuristic = c(5L, 6L),
  warn = TRUE,
  train = TRUE,
  na = na_patterns,
  range_value = c("start", "end", "na"),
  range_sep = c("-", "to", ","),
  ...
)
```

Arguments

`x`	A vector of `character` dates, `Date`s, or `POSIXt`s
`force`	Whether to force conversion to `Date` (`force = "dt"`) or `POSIXct` (`force = "dttm"`). The default is no forcing (`force = "none"`).
`orders`	A `character` vector of date-time formats. Each order string is a series of formatting characters as listed in `base::strptime()` but might not include the "%" prefix. For example, "ymd" will match all the possible dates in year, month, day order. Formatting orders might include arbitrary separators. These are discarded. See details of `lubridate::parse_date_time()` for the implemented formats. If multiple order strings are supplied, the order of applied formats is determined by the `select_formats` parameter in `lubridate::parse_date_time()` (if passed via dots).
`tz_heuristic`	Hours to consider in determining presence of sub-daily information. Only exact hours (i.e. 5:00:00) will be combined. The default corresponds to accidental encoding of the CST-UTC offset as hours.
`warn`	Should warnings be thrown when necessary? `FALSE` will suppress all warnings in the conversion process.

train	logical, default TRUE. Whether to train formats on a subset of the input vector. The result of this is that supplied orders are sorted according to performance on this training set, which commonly results in increased performance. Please note that even when train = FALSE (and exact = FALSE, if passed via dots) guessing of the actual formats is still performed on a pseudo-random subset of the original input vector. This might result in All formats failed to parse error.See notes in lubridate::parse_date_time().
na	Regular expressions to convert to NA
range_value	The value to use if the date is given as a range; can be the start date, the end date, or fill with NA
range_sep	Separators used for date ranges
...	Additional arguments to pass to convert_to_datetime(). These will, in turn, be passed to further methods, including excel_numeric_to_date(), parse_date_time(), and as.POSIXct().

Value

A Date or POSIXct vector

`std_intvl` *Standardize Interval Representations*

Description

`std_invl()` standardizes the various representations of numeric intervals found in the ML in HCT dataset. These intervals are assumed to be in percentage values and thus lie between 0 and 100. Explicit intervals with upper and lower bounds, as well as implicit intervals using < and >, are handled (<= and >= are currently not supported). The return value simplifies to </>/<=/>= or a single numeric value if possible and uses standard interval notation if not.

Usage

```
std_intvl(
  x,
  less_than = c("LESS THAN",
    "[A-Z ]*NOTHING TO SUGGEST[A-Z ]*SENSITIVITY[A-Z ]*(?=[0-9])"),
  greater_than = c("GREATER THAN"),
  na = na_patterns,
  std_chr = TRUE,
  warn = TRUE,
  ...
)
```

Arguments

x	A `character` vector
less_than	Regex patterns to consider `"<"`. Passed to `stringr::str_replace()`. Can be a vector of patterns.
greater_than	Regex patterns to consider `">"`. Passed to `stringr::str_replace()`. Can be a vector of patterns.
na	Regex patterns to consider `NA`. Passed to `stringr::str_detect()`. Can be a vector of patterns.
std_chr	Whether to standarize the strings before parsing
warn	Whether to emit a warning when potential numeric values are not able to be converted to an interval
...	Arguments passed on to chr_to_num `std` Whether to standardize the vector before cleaning and converting `convert` Whether to actually convert to `numeric` `replace` A `data.frame` of regular expressions and strings to replace them; regular expression should be in a column named `pattern`, and replacements should be in a column named `replacement`. Each row is passed to `stringr::str_replace()`. `per_action` How to treat %/percent/per million/etc labels. `drop` simply removes the labels, `divide` divides the value by the appropriate denominator, and `ignore` does nothing. `multiple_decimals` How to handle multiple decimals within a number `donor_host` Which value to use when values for both a donor and a host are given

Value

A `character` vector

---

std_lgl *Standardize* `logical` *Representations in Various Formats*

Description

`std_lgl()` converts other classes to `logical` vectors. All but `character` use `as.logical()`; `character` vectors are converted by first (optionally) standardizing with `std_chr` and then assigning logical value based on the regular expression in `true`, `false`, and `na`.

Usage

```
std_lgl(
  x,
  true = c("^TRUE$", "^1$", "^YES", "^POS", "^ALIVE", "^ON THERAPY"),
  false = c("^FALSE$", "^0$", "^NO", "^NEG", "^EXPIRED", "^DECEASED", "^OFF THERAPY"),
  na = na_patterns,
  std_chr = TRUE,
  warn = TRUE
)
```

Arguments

x	A vector to convert
true	Regex patterns to consider TRUE. Passed to stringr::str_detect(). Can be a vector of patterns.
false	Regex patterns to consider FALSE. Passed to stringr::str_detect(). Can be a vector of patterns.
na	Regex patterns to consider NA. Passed to stringr::str_detect(). Can be a vector of patterns.
std_chr	Whether to standardized a character vector before parsing
warn	Whether to warn if character strings were not converted to logical

Value

A logical vector

---

std_num *Convert and Standardize Numeric Values in Various Forms*

Description

std_num() converts all base classes, as well as int64, factor, Date, and POSIXt vectors to the simplest numeric form possible.

Usage

```
std_num(x, na = na_patterns, std_chr = TRUE, warn = TRUE, ...)
```

Arguments

x	A vector to convert to numeric
na	Regex patterns to consider NA. Passed to stringr::str_detect(). Can be a vector of patterns.
std_chr	Whether to standardize a character or factor before conversion
warn	Whether to warn when strings cannot be converted; passed to chr_to_num()

... Arguments passed on to chr_to_num

std Whether to standardize the vector before cleaning and converting

convert Whether to actually convert to numeric

replace A data.frame of regular expressions and strings to replace them; regular expression should be in a column named pattern, and replacements should be in a column named replacement. Each row is passed to stringr::str_replace().

per_action How to treat %/percent/per million/etc labels. drop simply removes the labels, divide divides the value by the appropriate denominator, and ignore does nothing.

multiple_decimals How to handle multiple decimals within a number

donor_host Which value to use when values for both a donor and a host are given

Details

character vectors are standardized using std_chr() by default, then converted. factors are treated as character vectors, rather than using the underlying integer representation. double and int64 vectors will be converted to integer if this does not cause overflow or loss of precision. Date is converted to integer, and POSIXt is converted to integer if the range allows, otherwise double.

Value

A numeric vector

Index