Package: rxref (via r-universe)

June 2, 2026

Title Tidy Utilities for 'RxNorm' and NDC Resolution

Version 0.4.0

Description Provides a tidy, vectorized interface to the 'RxNorm' / 'RxNav' API for resolving drug names, 'RxNorm' Concept Unique Identifiers (RxCUIs), National Drug Codes (NDCs), and related drug concept metadata. The package supports workflows for mapping between drug names, RxCUIs, NDCs, ingredients, products, drug classes, and related concepts using data from the U.S. National Library of Medicine's 'RxNav' services <`https://lhncbc.nlm.nih.gov/RxNav/APIs/`> and 'RxNorm' <`https://www.nlm.nih.gov/research/umls/rxnorm/`>.

License MIT + file LICENSE

URL `https://www.stevenmsmith.org/rxref/`, `https://github.com/ssmithm/rxref`

BugReports `https://github.com/ssmithm/rxref/issues`

Depends R (>= 4.1)

Imports cachem, cli, digest, dplyr, httr2 (>= 1.0.0), jsonlite, lifecycle, memoise, purrr, rlang, stringr, tibble, tidyr, vctrs

Suggests curl, knitr, rmarkdown, testthat (>= 3.0.0), withr

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.0.0

RoxygenNote 8.0.0

Config/pak/sysreqs libicu-dev libssl-dev

Repository https://ssmithm.r-universe.dev

Date/Publication 2026-06-02 05:37:25 UTC

RemoteUrl https://github.com/ssmithm/rxref

RemoteRef HEAD

RemoteSha 2304eef10100abf09aa6febb797ed7f31d2a5589

Contents

`default_product_ttys` *Default RxNorm product term types*

Description

Returns the default RxNorm term types used by rxref when identifying drug products.

Usage

```
default_product_ttys()
```

Details

The default set is intentionally focused on product-level concepts: semantic clinical drugs, semantic branded drugs, generic packs, and branded packs.

Value

A character vector of RxNorm term type abbreviations.

Examples

```
default_product_ttys()
```

`exclude_products_with_ingredients` *Exclude products containing specified ingredients*

Description

Remove all rows for products that contain one or more user-specified ingredients. This is useful when broad ingredient-based product searches return fixed-dose combination products that contain an ingredient of interest but are not clinically relevant to a specific study definition.

Usage

```
exclude_products_with_ingredients(
  data,
  ingredients = NULL,
  ingredient_rxcuis = NULL,
  product_id_col = "product_rxcui",
  ingredient_name_col = "ingredient_name",
  ingredient_rxcui_col = "ingredient_rxcui",
  return_excluded = FALSE
)
```

Arguments

`data`	A data frame containing product RxCUIs and ingredient information, such as that produced by `products_for_ingredients()`.
`ingredients`	Optional character vector of ingredient names to exclude. These are resolved to ingredient RxCUIs using `find_ingredients()`.
`ingredient_rxcuis`	Optional character vector of ingredient RxCUIs to exclude. These are used directly and do not require name resolution.
`product_id_col`	Name of the column containing product RxCUIs. Default is `product_rxcui`.
`ingredient_name_col`	Name of the column containing ingredient names. Used for reporting and, when needed, fallback matching. Default is `ingredient_name`.
`ingredient_rxcui_col`	Name of the column containing ingredient RxCUIs. Default is `ingredient_rxcui`.
`return_excluded`	Logical. If `FALSE`, return the filtered data frame. If `TRUE`, return a list with `data`, `excluded`, and `resolved_ingredients`.

Details

Exclusions are applied at the product level. If any ingredient in a product matches the exclusion list, all rows for that product are removed.

Value

If `return_excluded = FALSE`, a data frame of the same general shape as `data`, with excluded product rows removed.

If `return_excluded = TRUE`, a list with:

data The filtered data frame.

excluded Rows from the input data belonging to excluded products.

resolved_ingredients The ingredient names/RxCUIs used for exclusion.

Examples

```
## Not run:
antihtn_clean <- antihtn_products |>
  exclude_products_with_ingredients(ingredients = "sacubitril")

antihtn_audit <- antihtn_products |>
  exclude_products_with_ingredients(
    ingredients = "sacubitril",
    return_excluded = TRUE
  )

antihtn_audit$data
antihtn_audit$excluded
antihtn_audit$resolved_ingredients
```

```
## End(Not run)
```

`extended_product_ttys` *Extended RxNorm product term types*

Description

Returns an extended set of RxNorm term types that includes the default product term types plus dose-form and dose-form-group concepts.

Usage

```
extended_product_ttys()
```

Details

This can be useful when a broader set of product-related RxNorm concepts is desired.

Value

A character vector of RxNorm term type abbreviations.

Examples

```
extended_product_ttys()
```

`extended_ttys` *Extended RxNorm term types*

Description

Returns an extended set of RxNorm term types that includes the default and extended product term types plus drug component, brand name, and ingredient concepts. This list includes essentially all term types that capture a specific ingredient, thus it excludes dose form, dose form group, prescribable name, synonyms, and others that are not associated with a specific ingredient.

Usage

```
extended_ttys()
```

Details

This can be useful when the broadest set of RxNorm concepts that still capture an ingredient is desired.

Value

A character vector of RxNorm term type abbreviations.

Examples

```
extended_ttys()
```

`filter_products_by_route`

Filter RxNorm product concepts by route of administration.

Description

Filters a product-level rxref table using route information from `get_clinical_attributes()`. Useful when a drug or class includes products across multiple routes, such as oral tablets, ophthalmic solutions, injectables, patches, or topical formulations.

Usage

```
filter_products_by_route(
  products,
  route = "ORAL",
  keep_route_info = TRUE,
  include_historical = FALSE,
  show_progress = interactive()
)
```

Arguments

`products`	A tibble containing a `product_rxcui` column, such as the output of `products_for_ingredients()`.
`route`	Character vector of routes to keep. Common values include `"ORAL"`, `"INJECTION"`, `"OPHTHALMIC"`, `"OTIC"`, `"INHALATION"`, `"TOPICAL"`, `"RECTAL"`, `"VAGINAL"`, `"URETHRAL"`, and `"IMPLANT"`.
`keep_route_info`	Logical. If `TRUE` (default), append route, dose form, and dose-form group summaries to the returned table.
`include_historical`	Logical. If `TRUE`, use RxCUI history status metadata as a fallback for RxCUIs that do not return active clinical attributes. This is useful for obsolete, remapped, quantified, or otherwise non-current RxCUIs found in historical prescribing data. If `products` contains a `concept_status` column with non-`"active"` value, historical lookup is enabled automatically.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble containing only rows whose `product_rxcui` has at least one matching route.

Examples

```
## Not run:
ing <- find_ingredients("metoprolol")

prods <- products_for_ingredients(
  ing$rxcui,
  include_combos = TRUE
)

filter_products_by_route(prods, route = "ORAL")

# Historical/current products can also be route-filtered
prods_hist <- products_for_ingredients(
  ing$rxcui,
  include_combos = TRUE,
  concept_status = "active_and_historical"
)

filter_products_by_route(prods_hist, route = "ORAL")

## End(Not run)
```

`find_classes` *Find RxClass drug classes by class name*

Description

Find RxClass drug classes by class name

Usage

```
find_classes(query, class_types = NULL)
```

Arguments

`query`	Character string to search for.
`class_types`	Optional class type filter, such as `"EPC"`, `"ATC1"`, `"ATC2"`, `"ATC3"`, `"ATC4"`, `"CHEM"`, `"DISEASE"`, or `"VA"`.

Value

A tibble of matching class concepts.

`find_ingredients` *Resolve a free-text drug name to ingredient CUIs (IN/PIN)*

Description

Uses RxNav approximateTerm, then fetches properties for each candidate and filters to ingredient-type concepts (TTY `IN` and, optionally, `PIN`).

Usage

```
find_ingredients(
  term,
  max_entries = 10,
  include_pin = TRUE,
  show_progress = interactive()
)
```

Arguments

`term`	Character vector (free text).
`max_entries`	Integer. Max approximate-term candidates per input (default 10).
`include_pin`	Logical. Include precise-ingredient (`PIN`) CUIs (default TRUE).
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble with columns: `input`, `rxcui`, `name`, `tty`, `score`.

Examples

```
if (identical(Sys.getenv("RXREF_ONLINE"), "1")) {
  find_ingredients("metformin")
}
```

`get_atc` *Get ATC classes for RxNorm drugs*

Description

`get_atc()` is a convenience wrapper for `get_classes()` that returns Anatomical Therapeutic Chemical (ATC) class assertions for RxNorm drugs.

Usage

```
get_atc(
  x,
  by = c("rxcui", "name"),
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble of ATC class assertions.

`get_atcprod` *Get ATC product-level classes for RxNorm drugs*

Description

`get_atc()` is a convenience wrapper for `get_classes()` that returns Anatomical Therapeutic Chemical (ATC) product-level class assertions for RxNorm drugs.

Usage

```
get_atcprod(
  x,
  by = c("rxcui", "name"),
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble of ATCPROD class assertions.

`get_chemical_structure` *Get chemical-structure assertions*

Description

`get_chemical_structure()` is a convenience wrapper for `get_classes()` that returns chemical-structure assertions for RxNorm drugs.

Usage

```
get_chemical_structure(
  x,
  by = c("rxcui", "name"),
  rela_source = c("FDASPL", "DAILYMED", "MEDRT"),
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`rela_source`	Relationship source. Defaults to FDASPL, DAILYMED, and MEDRT.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble of chemical-structure assertions.

`get_class_members` *Get RxNorm members of an RxClass class*

Description

`get_class_members()` queries the RxClass API to identify members of a specific drug class. Typically, this would be used to identify all ingredient RxCUIs in a class, which can then be combined with other search functions to extract related product RxCUIs or NDCs.

Usage

```
get_class_members(
  class_id,
  rela_source,
  rela = NULL,
  include_indirect = TRUE,
  ttys = NULL
)
```

Arguments

`class_id`	RxClass class identifier.
`rela_source`	RxClass relationship source, such as `"ATC"`, `"ATCPROD"`, `"DAILYMED"`, `"FDASPL"`, `"MEDRT"`, `"SNOMEDCT"`, or `"VA"`.
`rela`	Optional relationship filter.
`include_indirect`	Logical; if `TRUE`, include direct and indirect members. If `FALSE`, return direct members only.
`ttys`	Optional RxNorm term type filter, such as `"IN"`, `"PIN"`, `"SCD"`, or `"SBD"`.

Value

A tibble of RxNorm drug members.

`get_classes` *Get RxClass assertions for RxNorm drugs*

Description

[Experimental]

`get_classes()` returns RxClass assertions for RxNorm concepts or drug names. These include drug-class relationships from sources such as ATC, FDASPL, DAILYMED, MEDRT, SNOMEDCT, and VA. Depending on the source and relationship, returned assertions may represent pharmacologic classes, chemical structures, mechanisms of action, physiologic effects, indications, contraindications, VA classes, ATC classes, or SNOMED CT dispositions/structures.

Because different sources use different classification logic, this function preserves class type, relationship, and relationship source rather than collapsing results into a single class label.

Usage

```
get_classes(
  x,
  by = c("rxcui", "name"),
  rela_source = NULL,
  relas = NULL,
  class_types = NULL,
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`rela_source`	Optional RxClass relationship source filter, such as `"ATC"`, `"ATCPROD"`, `"DAILYMED"`, `"FDASPL"`, `"MEDRT"`, `"SNOMEDCT"`, or `"VA"`. May be a character vector; values are queried separately.
`relas`	Optional relationship filter. Examples include `"has_EPC"`, `"has_MoA"`, `"has_PE"`, `"may_treat"`, `"ci_with"`, `"has_chemical_structure"`, and others supported by RxClass.
`class_types`	Optional filter on returned class types, such as `"EPC"`, `"MOA"`, `"PE"`, `"DISEASE"`, `"CHEM"`, `"VA"`, or `"ATC1-4"`.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble with one row per RxClass assertion.

`get_clinical_attributes` *Clinical attributes from the concept (SCD/SBD) or related SCD/SBD*

Description

If `rxcui` is already a clinical drug (TTY = SCD or SBD), parse attributes directly from its own name. Otherwise, query `/rxcui/{id}/related?tty=SCD,SBD` and parse.

Usage

```
get_clinical_attributes(
  rxcui,
  include_historical = FALSE,
  show_progress = interactive()
)
```

Arguments

`rxcui`	Character vector of RxCUIs
`include_historical`	Logical. If `TRUE`, use RxCUI history status metadata as a fallback for RxCUIs that do not return active clinical attributes. This is useful for obsolete, remapped, quantified, or otherwise non-current RxCUIs found in historical prescribing data.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Details

In addition to strength and dose form, this returns route, dose-form group (DFG), brand/generic flags, ingredient summaries, and a simple active/inactive status.

Note there is a fair amount of parsing of the RxNorm STR value to try to extract relevant information (e.g., strength, dose_form), so check closely before trusting. There may be edge cases that are not correctly parsed.

For combination products, ingredient-related columns may contain multiple semicolon-delimited values, such as `"amlodipine; valsartan"`.

Value

A tibble with columns:

rxcui Input RxCUI

related_rxcui Clinical drug RxCUI (SCD/SBD) used for attributes

name Clinical drug name

tty Term type (SCD/SBD, etc.)

strength Parsed strength string (e.g. "500 MG")

dose_form Parsed dose form (e.g. "Extended Release Oral Tablet")

route Route parsed from dose form / DFG (e.g. "ORAL", "INJECTION")

dose_form_group Dose form group (DFG), if available

is_brand Logical; TRUE for branded clinical concepts (SBD/BPCK)

is_generic Logical; TRUE for generic clinical concepts (SCD/GPCK)

ingredient_count Number of distinct ingredients

ingredient_rxcui Ingredient RxCUI. For combination products, multiple values are returned as semicolon-delimited strings.

ingredient_name Ingredient name. For combination products, multiple values are returned as semicolon-delimited strings.

ingredient_tty Ingredient term type (IN/PIN/MIN). For combination products, multiple values are returned as semicolon-delimited strings.

is_multi_ingredient Logical; TRUE if >1 ingredient

suppress Raw RxNorm suppress flag from properties

status Simple status derived from suppress: "ACTIVE" vs "INACTIVE"

Examples

```
## Not run:
get_clinical_attributes(c("861007","860975")) |>
  dplyr::select(rxcui, related_rxcui, name, strength, dose_form, route, tty) |>
  head()

## End(Not run)
```

`get_contraindications` *Get contraindication assertions from MED-RT*

Description

`get_contraindications()` is a convenience wrapper for `get_medrt()` that returns only contraindication MED-RT assertions for RxNorm drugs.

Usage

```
get_contraindications(
  x,
  by = c("rxcui", "name"),
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble of MED-RT contraindication assertions.

`get_drug_classes` *Get class-like RxClass assertions for RxNorm drugs*

Description

[Experimental]

`get_drug_classes()` is an experimental convenience function that returns RxClass assertions that are likely to behave like drug-class labels.

It combines selected assertions from ATC, ATCPROD, FDA/SPL EPC, VA, and SNOMED CT disposition relationships. It intentionally excludes relationship types that usually describe contraindications, indications, physiologic effects, chemical structures, or other non-class assertions.

For a more compact and ingredient-oriented output, rely on defaults for include_sources. For extended list of product-level sources, consider adding "ATCPROD" and "VA".

This function is experimental because "drug class" is not a single native RxClass concept. Different sources use different classification logic, and this helper applies an opinionated filter to return class-like assertions. For source-specific results, use `get_classes()`, `get_atc()`, `get_epc()`, `get_va()`, or related functions directly.

Usage

```
get_drug_classes(
  x,
  by = c("rxcui", "name"),
  include_sources = c("ATC", "EPC", "SNOMEDCT"),
  collapse = TRUE,
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`include_sources`	Character vector of class-like sources to include. Defaults to `c("ATC", "ATCPROD", "EPC", "VA", "SNOMEDCT")`.
`collapse`	Logical; if `TRUE`, returns one row per unique class assertion per input and source, dropping drug-specific columns. If `FALSE`, returns the full source-specific rows, including matched RxCUI, drug name, and term type.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble of class-like RxClass assertions.

`get_epc` *Get FDA/SPL established pharmacologic classes*

Description

`get_epc()` is a convenience wrapper for `get_classes()` that returns FDA labeling class assertions for RxNorm drugs.

Usage

```
get_epc(
  x,
  by = c("rxcui", "name"),
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble of established pharmacologic class assertions.

`get_may_treat` *Get indication/treatment assertions from MED-RT*

Description

`get_may_treat()` is a convenience wrapper for `get_medrt()` that returns indications (disease assertions) from MED-RT for RxNorm drugs, when available. Note that absence of a row (or an empty table altogether) should not necessarily be interpreted as absence of an indication.

Usage

```
get_may_treat(
  x,
  by = c("rxcui", "name"),
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble of MED-RT may-treat assertions.

`get_medrt` *Get MED-RT assertions for RxNorm drugs*

Description

`get_medrt()` is a convenience wrapper for `get_classes()` that returns MED-RT assertions for RxNorm drugs. These include, for example, asserted mechanisms of action, contraindications, physiologic effects (including adverse side effects), etc.

Usage

```
get_medrt(
  x,
  by = c("rxcui", "name"),
  relas = NULL,
  class_types = NULL,
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`relas`	Optional MED-RT relationship filter.
`class_types`	Optional MED-RT class type filter.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble of MED-RT assertions.

`get_medrt_moa` *Get MED-RT mechanism-of-action assertions*

Description

`get_medrt_moa()` is a convenience wrapper for `get_medrt()` that returns MED-RT mechanism-of-action assertions when available. Not all drugs have MED-RT has_moa assertions; for some drugs, mechanism-like or class-like information may instead be available from FDA/SPL EPC via `get_epc()` or as physiologic effects via `get_medrt_pe()`.

Usage

```
get_medrt_moa(
  x,
  by = c("rxcui", "name"),
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble of MED-RT mechanism-of-action assertions.

`get_medrt_pe` *Get MED-RT physiologic-effect assertions*

Description

`get_medrt_pe()` is a convenience wrapper for `get_medrt()` that returns only physiologic effect MED-RT assertions for RxNorm drugs.

Usage

```
get_medrt_pe(
  x,
  by = c("rxcui", "name"),
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble of MED-RT physiologic-effect assertions.

`get_properties` *Get core RxNorm properties for one or more RxCUIs*

Description

Get core RxNorm properties for one or more RxCUIs

Usage

```
get_properties(rxcui, show_progress = interactive())
```

Arguments

`rxcui`	Character vector of RxCUIs
`show_progress`	Logical. Show a progress bar in interactive sessions.

Value

tibble with rxcui, name, synonym, tty, language, suppress, umlscui.

Examples

```
if (identical(Sys.getenv("RXREF_ONLINE"), "1")) {
get_properties(c("860975","1049630"))
}
```

`get_va` *Get VA drug classes for RxNorm drugs*

Description

`get_va()` is a convenience wrapper for `get_classes()` that returns VA drug class assertions for RxNorm drugs.

Usage

```
get_va(
  x,
  by = c("rxcui", "name"),
  extended = TRUE,
  keep_input = TRUE,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`extended`	Logical; if `TRUE`, includes extended VA class assertions.
`keep_input`	Logical; if `TRUE`, includes the original input value.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble of VA class assertions.

`ingredients_for_rxcui` *Get ingredient concepts for RxCUIs*

Description

Maps one or more RxCUIs to related ingredient concepts, returning ingredient RxCUIs, names, and term types. This is useful when the input is already a specific RxNorm product concept rather than a free-text drug name.

Usage

```
ingredients_for_rxcui(
  rxcui,
  include_pin = TRUE,
  include_min = FALSE,
  show_progress = interactive()
)
```

Arguments

`rxcui`	Character vector of RxCUIs.
`include_pin`	Logical. Include precise ingredient concepts (`PIN`).
`include_min`	Logical. Include multiple ingredient concepts (`MIN`).
`show_progress`	Logical. Show a progress bar in interactive sessions.

Value

A tibble with columns `rxcui`, `ingredient_rxcui`, `ingredient_name`, and `ingredient_tty`.

`map_ndc_to_rxcui` *Map NDCs to RxCUIs*

Description

Convenience wrapper around `map_to()` for mapping National Drug Codes (NDCs) to RxNorm Concept Unique Identifiers (RxCUIs).

Usage

```
map_ndc_to_rxcui(x, show_progress = interactive())
```

Arguments

`x`	Character vector of NDCs.
`show_progress`	Logical; if `TRUE`, show a progress bar for vectorized mapping operations. Defaults to `interactive()`.

Details

NDCs may be supplied as 10-digit, 11-digit, or hyphenated values. Input NDCs are normalized to 11-digit format before querying RxNorm.

Value

A tibble with columns:

input Original input NDC.

ndc11 Normalized 11-digit NDC.

rxcui Mapped RxCUI, if found.

Examples

```
## Not run:
map_ndc_to_rxcui("00093-7424-56")

## End(Not run)
```

`map_rxcui_to_ndc` *Map RxCUIs to NDCs*

Description

Convenience wrapper around `map_to()` for mapping RxNorm Concept Unique Identifiers (RxCUIs) to National Drug Codes (NDCs).

Usage

```
map_rxcui_to_ndc(
  x,
  status = NULL,
  history = c("active", "direct", "all"),
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs.
`status`	Optional character vector of NDC statuses to retain, such as `"ACTIVE"`, `"OBSOLETE"`, `"ALIEN"`, or `"UNKNOWN"`. If `NULL`, no status-based filtering is applied to the retrieved NDCs. Note that `history = "active"` uses RxNorm's active NDC endpoint, so returned NDCs are expected to be active even when `status = NULL`.
`history`	For `to = "ndc"` only, the NDC association history to retrieve. One of `"active"`, `"direct"`, or `"all"`. • `"active"` retrieves currently active NDCs associated with the supplied RxCUI. • `"direct"` retrieves NDCs ever directly associated with the supplied RxCUI. • `"all"` retrieves NDCs ever directly or indirectly associated with the supplied RxCUI, including associations through remapped or archived concepts. If `NULL` and `to = "ndc"`, defaults to `"active"`. Ignored when `to = "rxcui"`.
`show_progress`	Logical; if `TRUE`, show a progress bar for vectorized mapping operations. Defaults to `interactive()`.

Details

By default, this function retrieves currently active NDCs associated with the supplied RxCUIs. Use `history = "direct"` or `history = "all"` to retrieve historical NDC associations.

`status = NULL` means that no status filter is applied after NDCs are retrieved. It does not, by itself, request historical NDCs. To retrieve historical NDCs, set `history = "direct"` or `history = "all"`.

Value

A tibble with one row per RxCUI/NDC mapping.

For `history = "active"`, returns columns:

rxcui Input RxCUI.

ndc11 Mapped NDC.

ndc_status NDC status returned by RxNorm, when available.

For `history = "direct"` or `"all"`, the output may also include:

related_rxcui RxCUI associated with the historical NDC record. This may differ from the input RxCUI when historical or indirect associations are retrieved.

ndc_start_date Start date of the NDC association, when available.

ndc_end_date End date of the NDC association, when available.

Examples

```
## Not run:
# Current active NDCs
map_rxcui_to_ndc("1049630")

# NDCs ever directly associated with this RxCUI
map_rxcui_to_ndc("1049630", history = "direct")

# All historical NDCs, retaining obsolete NDCs only
map_rxcui_to_ndc(
  "1049630",
  history = "all",
  status = "OBSOLETE"
)

## End(Not run)
```

`map_to` *Map between NDCs and RxCUIs*

Description

`map_to()` maps identifiers between National Drug Codes (NDCs) and RxNorm Concept Unique Identifiers (RxCUIs).

Usage

```
map_to(
  x,
  to = c("rxcui", "ndc"),
  status = NULL,
  history = NULL,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of NDCs or RxCUIs.
`to`	Direction of mapping. One of `"rxcui"` or `"ndc"`. Use `"rxcui"` to map NDCs to RxCUIs and `"ndc"` to map RxCUIs to NDCs.
`status`	For `to = "ndc"` only, optional character vector of NDC statuses to retain, such as `"ACTIVE"`, `"OBSOLETE"`, `"ALIEN"`, or `"UNKNOWN"`. If `NULL`, no status-based filtering is applied to the retrieved NDCs. Note that when `history = "active"`, RxNorm's active NDC endpoint is used, so returned NDCs are expected to be active even when `status = NULL`.
`history`	For `to = "ndc"` only, the NDC association history to retrieve. One of `"active"`, `"direct"`, or `"all"`. • `"active"` retrieves currently active NDCs associated with the supplied RxCUI. • `"direct"` retrieves NDCs ever directly associated with the supplied RxCUI. • `"all"` retrieves NDCs ever directly or indirectly associated with the supplied RxCUI, including associations through remapped or archived concepts. If `NULL` and `to = "ndc"`, defaults to `"active"`. Ignored when `to = "rxcui"`.
`show_progress`	Logical; if `TRUE`, show a progress bar for vectorized mapping operations. Defaults to `interactive()`.

Details

When mapping from NDC to RxCUI, NDCs may be supplied as 10-digit, 11-digit, or hyphenated values. Input NDCs are normalized to 11-digit format before querying RxNorm.

When mapping from RxCUI to NDC, the `history` argument controls whether only currently active NDCs are retrieved or whether historical NDC associations are also included.

RxNorm distinguishes between currently active NDCs and historical NDC associations. The default behavior, `history = "active"`, uses RxNorm's active NDC endpoint and preserves the earlier behavior of `map_to()`.

To retrieve obsolete, discontinued, or otherwise historical NDCs, use `history = "direct"` or `history = "all"`. Setting `status = NULL` does not by itself request historical NDCs; it only means that no status filter is applied after NDCs are retrieved.

Value

A tibble.

For `to = "rxcui"`, returns one row per input NDC/RxCUI mapping with columns:

input Original input NDC.

ndc11 Normalized 11-digit NDC.

rxcui Mapped RxCUI, if found.

For `to = "ndc"` and `history = "active"`, returns one row per RxCUI/NDC mapping with columns:

rxcui Input RxCUI.

ndc11 Mapped NDC.

ndc_status NDC status returned by RxNorm, when available.

For `to = "ndc"` and `history` equal to `"direct"` or `"all"`, the output may also include:

related_rxcui RxCUI associated with the historical NDC record. This may differ from the input RxCUI when historical or indirect associations are retrieved.

ndc_start_date Start date of the NDC association, when available.

ndc_end_date End date of the NDC association, when available.

Examples

```
## Not run:
# Map an NDC to RxCUI
map_to("00093-7424-56", to = "rxcui")

# Map an RxCUI to currently active NDCs
map_to("1049630", to = "ndc")

# Map an RxCUI to all directly associated historical NDCs
map_to("1049630", to = "ndc", history = "direct")

# Map an RxCUI to all historical NDCs and retain obsolete NDCs only
map_to("1049630", to = "ndc", history = "all", status = "OBSOLETE")

## End(Not run)
```

`ndc_status` *Get RxNav status for NDCs*

Description

`ndc_status()` requests the status of an NDC entry for one or more user-supplied NDC numbers. The returned status will be one "ACTIVE", "OBSOLETE", or "UNSPECIFIED". On occasion, a NULL value may be pulled (resulting in an `NA` in the returned tibble), when RxNorm has no status for a given NDC, or if an invalid NDC is supplied.

Usage

```
ndc_status(ndc, show_progress = interactive())
```

Arguments

`ndc`	character vector; hyphenated or digits
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble with three columns: ndc_input, ndc11, ndc_hyph, status

---

`product_ttys` *RxNorm product term type sets*

Description

Returns common RxNorm term type sets used by rxref.

Usage

```
product_ttys(set = c("default", "extended_product", "extended"))
```

Arguments

`set`	One of `"default"`, `"extended_product"`, or `"extended"`.

Value

A character vector of RxNorm term type abbreviations.

Examples

```
product_ttys()
product_ttys("extended")
```

`products_for_ingredients`

Expand ingredient CUIs to product CUIs that truly contain the ingredient

---

Description

Tries multiple RxNav endpoints and verifies candidates truly contain the queried ingredient or one of its acceptable related ingredient concepts. The function unions candidates from multiple sources, verifies ingredient containment, and reports the number of ingredients represented by each product concept.

Usage

```
products_for_ingredients(
  ingredient_rxcui,
  ttys = .rxref_default_ttys,
  route = NULL,
  include_combos = TRUE,
  concept_status = c("active", "active_and_historical"),
  historical_status = c("Obsolete", "Remapped", "Quantified", "NotCurrent"),
  show_progress = interactive()
)
```

Arguments

`ingredient_rxcui`	Character vector of ingredient CUIs (TTY `IN` or `PIN`).
`ttys`	Character vector of TTYs to include. Defaults to product-facing TTYs returned by `product_ttys()`. Pass a larger set if you want groups, components, branded concepts, or other product-related concepts, for example `product_ttys("extended_product")`.
`route`	Optional character vector of routes to retain. If `NULL`, no route filtering is performed. Route filtering uses `get_clinical_attributes()`. Route filtering is intended for product-level TTYs and may not filter well on broader group or package TTYs.
`include_combos`	Logical. If `FALSE`, keep only single-ingredient products, where ingredient count is based on distinct `IN` concepts when available and otherwise falls back to distinct `PIN` concepts.
`concept_status`	Character. Which RxNorm concept universe to search. `"active"` uses active-scope RxNav relationship endpoints and is the default. `"active_and_historical"` also searches historical RxNorm concepts using all-status concept retrieval and RxCUI history status metadata. Historical searching is slower and is intended for mapping older prescribing or dispensing data.
`historical_status`	Character vector of historical RxNorm statuses to include when `concept_status = "active_and_historical"`. Defaults to `c("Obsolete", "Remapped", "Quantified", "NotCurrent")`. These values use RxNorm status definitions:

- `"Obsolete"` The concept is obsolete in the current RxNorm data set, and RxNorm has not designated an active concept as equivalent.
- `"Remapped"` The concept was active or obsolete at one time, is no longer in the current data set, and has been remapped to one or more active or obsolete concepts.
- `"Quantified"` The concept has been designated as non-dispensable because it lacks a quantity factor; related concepts with quantity factors may be available.
- `"NotCurrent"` The concept either exists in the current data set without RxNorm vocabulary terms, or existed in a previous monthly release but has since been removed and not remapped.

See the RxNorm API documentation for concept status values: https://lhncbc.nlm.nih.gov/RxNav/APIs/api-RxNorm.getAllConceptsByStatus.html.

`show_progress`	Logical. Show progress bars for long-running API retrieval, product matching, and optional route filtering steps. Defaults to `interactive()`.

Value

A tibble with one row per matched ingredient/product concept pair. For `concept_status = "active"`, columns include `ingredient_rxcui`, `product_rxcui`, `name`, `tty`, and `n_ingredients`. When `concept_status = "active_and_historical"`, additional columns include `concept_status`, `active_start_date`, `active_end_date`, `release_start_date`, and `release_end_date`.

`resolve` *Resolve free text, RxCUI, or NDC to RxCUI and preferred name*

Description

Vectorized over `x`. For free text, uses RxNorm approximateTerm. For NDC, uses findRxcuiById. For RxCUI, validates and returns properties.

Usage

```
resolve(
  x,
  type = c("auto", "name", "rxcui", "ndc"),
  max_entries = 1,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector: drug string, RxCUI, or NDC (10/11-digit or hyphenated)
`type`	One of c("auto","name","rxcui","ndc"). Default "auto" infers.
`max_entries`	Integer, passed to approximateTerm for name queries.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble with columns: input, type, rxcui, name, tty, score (if name), ndc11 (if ndc input), matched_term (if name input)

`rxclass_rela_sources` *Get RxClass relationship sources*

Description

Get RxClass relationship sources

Usage

```
rxclass_rela_sources()
```

Value

A tibble of RxClass relationship sources.

`rxclass_relas` *Get RxClass relationship types*

Description

`rxclass_relas()` is a helper function that queries the RxClass API for generally valid relationships within the data. Note that just because a relationship is generally valid within a rela_source, not every drug or RxCUI will have a corresponding relationship asserted.

Usage

```
rxclass_relas(rela_source)
```

Arguments

`rela_source`	RxClass relationship source, such as `"ATC"`, `"DAILYMED"`, `"FDASPL"`, `"MEDRT"`, `"SNOMEDCT"`, or `"VA"`.

Value

A tibble of relationship names for the selected source.

`rxclass_relationships` *Summarize RxClass relationship types for drugs*

Description

`rxclass_relationships()` is a diagnostic helper that evaluates the type(s) of relationship (rela) asserted in RxNorm MED-RT data and the number within each relationship asserted. In some cases, `get_medrt()` or its various convenience wrappers may return empty or shorter-than-expected tibbles because some relationships expected by the user are not asserted in RxNorm (for example, an MOA may not be asserted by MED-RT).

Usage

```
rxclass_relationships(
  x,
  by = c("rxcui", "name"),
  rela_source = NULL,
  show_progress = interactive()
)
```

Arguments

`x`	Character vector of RxCUIs or drug names.
`by`	One of `"rxcui"` or `"name"`.
`rela_source`	Optional RxClass relationship source filter.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.

Value

A tibble summarizing available relationship/class-type combinations.

`rxref_cache` *Manage the rxref cache*

Description

`rxref_cache()` returns the currently configured cache object. `rxref_cache_info()` returns a small summary of the current cache. `rxref_cache_clear()` clears cached RxNorm/RxClass API responses. `rxref_cache_use_memory()` switches rxref to an in-memory cache. `rxref_cache_use_disk()` switches rxref to an on-disk cache. `rxref_cache_disable()` uses an immediately expiring cache, effectively disabling cache reuse for the current R session.

Usage

```
rxref_cache()

rxref_cache_info()

rxref_cache_clear()

rxref_cache_use_memory(
  max_size = 200 * 1024^2,
  max_age = Inf,
  evict = c("lru", "fifo")
)

rxref_cache_use_disk(
  dir = rxref_cache_dir(),
  max_size = 1024^3,
  max_age = Inf,
  evict = c("lru", "fifo")
)

rxref_cache_disable()
```

Arguments

`max_size`	Maximum cache size in bytes.
`max_age`	Maximum age of cached objects, in seconds.
`evict`	Cache eviction policy passed to `cachem::cache_mem()`.
`dir`	Directory to use for an on-disk cache.

Details

rxref caches memoised API responses through a `cachem` cache object stored in `getOption("rxref.cache")`. By default, rxref uses an in-memory cache for the current R session.

Value

`rxref_cache()` returns a `cachem` cache object.

`rxref_cache_info()` returns a tibble with cache metadata.

`rxref_cache_clear()`, `rxref_cache_use_memory()`, `rxref_cache_use_disk()`, and `rxref_cache_disable()` invisibly return the configured cache object.

Examples

```
rxref_cache_info()

rxref_cache_clear()

rxref_cache_use_memory()
```

```
## Not run:
rxref_cache_use_disk()

## End(Not run)
```

`rxref_conf` *Configure rxref*

Description

Configure package-level settings used by rxref, including the RxNav base URL, API rate delay, and memoised API-response cache.

Usage

```
rxref_conf(
  base_url = getOption("rxref.base_url"),
  rate_delay = getOption("rxref.rate_delay"),
  cache = getOption("rxref.cache")
)
```

Arguments

`base_url`	Override the RxNav base URL (e.g., a local mirror)
`rate_delay`	Seconds to wait between HTTP calls
`cache`	A `cachem` cache object used by memoised API calls. See `rxref_cache()` for cache management helpers.

Value

A named list of current settings

`search_drug` *Search free-text drug name and return product CUIs and/or NDCs*

Description

High-level convenience: free text -> ingredient(s) (IN/PIN) -> verified product CUIs, and optionally expand to NDCs with status filtering.

Usage

```
search_drug(
  term,
  return = c("rxcui", "ndc", "both"),
  ndc_status = NULL,
  ttys = .rxref_default_ttys,
  route = NULL,
  show_progress = interactive(),
  ...
)
```

Arguments

`term`	Character vector; free-text drug names.
`return`	One of `c("rxcui","ndc","both")`. Note that `"both"` will return a list with both an rxcui tibble and an ndc tibble.
`ndc_status`	Optional character vector to filter NDCs. Options are "ACTIVE", "OBSOLETE", "UNSPECIFIED" (the API may also return no value, which will appear as NA).
`ttys`	Character vector of TTYs to include in product search. Defaults to `.rxref_default_ttys`. Other prespecified option is `.rxref_extended_ttys` or a character vector of explicit TTYs. Run `tty_catalogue()` to review options.
`route`	Optional character vector of routes to retain before returning products or mapping to NDCs. If `NULL`, no route filtering is performed. Common values include `"ORAL"`, `"INJECTION"`, `"OPHTHALMIC"`, `"INHALATION"`, and `"TOPICAL"`.
`show_progress`	Logical. Show a progress bar in interactive sessions. Progress is shown only when at least 5 inputs are supplied.
`...`	Passed to `products_for_ingredients()` (e.g., include_combos = FALSE)

Value

If `return="rxcui"`: tibble of products. If `"ndc"`: tibble of NDCs with `ingredient_rxcui`, `product_rxcui`, ndc11, `ndc_status`. If `"both"`: list(products=..., ndcs=...).

`tty_catalogue` *Catalogue of RxNorm TTY (Term Types)*

Description

Returns a tibble describing common RxNorm TTYs you may want to use when expanding ingredients to products. Includes whether each TTY typically maps cleanly to NDCs and whether it's included in the package's default or extended TTY sets.

Usage

```
tty_catalogue()
```

Details

Columns:

- `tty`: RxNorm term type code.
- `label`: Short, human-friendly name.
- `description`: What the TTY represents in RxNorm.
- `maps_to_ndc`: Logical; whether CUIs of this TTY usually map to NDCs via `/rxcui/{rxcui}/ndcs`.
- `typical_role`: How it's commonly used (product, component, group, name).
- `include_default`: Logical; included in `.rxref_default_ttys`.
- `include_extended`: Logical; included in `.rxref_extended_ttys`.

Value

A tibble with metadata for key TTYs.

Examples

```
tty_catalogue()

# TTYs that map to NDCs
subset(tty_catalogue(), maps_to_ndc)$tty

# See what your defaults and extended sets contain
subset(tty_catalogue(), include_default)$tty
subset(tty_catalogue(), include_extended)$tty

# Pick a custom set: products + components
with(tty_catalogue(),
     tty[tty %in% c("SCD","SBD","GPCK","BPCK","SCDC","SBDC")])
```

Index