

Package ‘ciecl’

October 2, 2026

Title International Classification of Diseases 'ICD-10'/'ICD-11' for Chile

Version 1.0.0

Description Tools for working with the International Classification of Diseases ('ICD-10' Chile official 'MINSAL'/'DEIS' v2018). Includes optimized 'SQL' search with 'SQLite', fuzzy matching of medical terms ('Jaro-Winkler'), Charlson and Elixhauser comorbidity calculation, 'WHO' 'ICD-11' 'API' integration, and hierarchical code validation. Data from Centro FIC Chile 'DEIS' <<https://deis.minsal.cl/centrofic/>>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports DBI, RSQLite, stringdist, stringr, dplyr, tibble, tools, utils, lifecycle, rlang, httr2, cli (>= 3.6.0)

Suggests comorbidity, gt, readxl, vcr (>= 2.0.0), withr, testthat (>= 3.3.0), knitr, rmarkdown

VignetteBuilder knitr

URL <https://docs.ropensci.org/ciecl/>,
<https://github.com/ropensci/ciecl>

BugReports <https://github.com/ropensci/ciecl/issues>

Config/testthat/edition 3

Config/Needs/website pkgdown, vcr

Language en, es

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Rodolfo Tasso Suazo [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-1264-4933>>, Grupo de Ciencia de Datos para Salud Pública, Escuela de Salud Pública, U. de Chile),
Francisca Jofre [ctb] (Package logo design),

Maëlle Salmon [rev] (Maëlle reviewed the package (v. 0.9.8) for
rOpenSci, see
<<https://github.com/ropensci/software-review/issues/765>>),
Yanina Bellini [rev] (Yanina reviewed the package (v. 0.9.8) for
rOpenSci, see
<<https://github.com/ropensci/software-review/issues/765>>)

Maintainer Rodolfo Tasso Suazo <rtasso@uchile.cl>

Repository CRAN

Date/Publication 2026-10-02 16:00:14 UTC

Contents

cie10_cl	2
cie10_clear_cache	4
cie10_disconnect	4
cie10_sql	5
cie11_search	6
cie_comorbid	8
cie_describe	9
cie_expand	10
cie_guide	10
cie_lookup	11
cie_map_comorbid	13
cie_norm	14
cie_search	15
cie_short	17
cie_table	18
cie_validate_vector	19
get_icd_api_key	19

Index	21
--------------	-----------

cie10_cl	<i>Dataset CIE-10 Chile oficial MINSAL/DEIS v2018</i>
----------	---

Description

Catálogo completo de diagnósticos CIE-10 en versión chilena (MINSAL/DEIS 2018), incluyendo categorías, subcategorías y metadatos jerárquicos. Contiene 39,877 filas que abarcan la totalidad de códigos del sistema de clasificación.

Usage

cie10_cl

Format

Un tibble con 39,877 filas y 11 columnas:

codigo character. Código CIE-10 alfanumérico (p. ej. "E11.0", "A00", "Z99.9"). Incluye tanto categorías de 3 caracteres como subcategorías con decimal.

descripcion character. Denominación del diagnóstico en español según terminología oficial MINSAL/DEIS.

categoria character. Código de categoría de 3 caracteres al que pertenece la fila (p. ej. "E11" para el código "E11.0"). Igual a codigo cuando la propia fila es una categoría.

seccion character. Rango de categorías que define la sección dentro del capítulo (p. ej. "E10-E14").

capitulo_nombre character. Nombre descriptivo del capítulo CIE-10 (p. ej. "Enfermedades endocrinas, nutricionales y metabólicas").

inclusion character. Términos de inclusión asociados al código, tal como figuran en el catálogo oficial. NA cuando no aplica.

exclusion character. Términos de exclusión (notas "Excl.") asociados al código. NA cuando no aplica.

capitulo character. Letra o rango que identifica el capítulo CIE-10 (valores "A" a "Z").

es_daga logical. TRUE si el código es daga (†), es decir, etiología en un par daga/asterisco. FALSE en caso contrario.

es_cruz logical. TRUE si el código es asterisco (*), es decir, manifestación en un par daga/asterisco. FALSE en caso contrario.

uso_cl character. Categoría de uso clínico del código según el catálogo MINSAL/DEIS: "principal", "legado", "causa_externa", "etiologico", o la combinación "causa_externa | principal". El valor "legado" identifica códigos sin uso clínico vigente en Chile.

Source

<https://deis.minsal.cl/centrofic/>

See Also

[cie_lookup\(\)](#), [cie_search\(\)](#), [cie10_sql\(\)](#)

Examples

```
data(cie10_cl)
head(cie10_cl)

# Códigos vigentes de uso clínico en Chile (excluye categoría "legado")
subset(cie10_cl, uso_cl != "legado") |> nrow()

# Capítulos disponibles
unique(cie10_cl$capitulo)
```

cie10_clear_cache	<i>Limpiar caché SQLite local (forzar rebuild)</i>
-------------------	--

Description

cie10 construye, en el primer uso, un archivo SQLite (cie10.db) a partir del dataset [cie10_cl](#) y lo guarda en una carpeta de datos del usuario (ver `tools::R_user_dir("cie10", "data")`). Esa "caché" evita reconstruir la base en cada sesión. Esta función la elimina y fuerza que la próxima consulta ([cie_search\(\)](#), [cie_lookup\(\)](#), [cie10_sql\(\)](#), etc.) la reconstruya desde cero.

No es necesario llamarla tras actualizar el paquete: la caché guarda la versión del paquete con que se construyó (tabla `cie10_meta`) y, si la versión cambió, se reconstruye automáticamente en el primer uso. Los casos en que conviene forzar el rebuild manual son: (1) se sospecha que el archivo `.db` está corrupto (errores de lectura SQL inesperados), o (2) se quiere liberar el espacio en disco que ocupa la caché.

Usage

```
cie10_clear_cache()
```

Value

Sin valor de retorno, se llama por sus efectos secundarios (elimina la caché SQLite).

See Also

[cie10_sql\(\)](#), [cie10_disconnect\(\)](#)

Other `sql_backend`: [cie10_disconnect\(\)](#), [cie10_sql\(\)](#)

Examples

```
# Ver ubicación de la caché
tools::R_user_dir("cie10", "data")

cie10_clear_cache() # Elimina cie10.db local
```

cie10_disconnect	<i>Cerrar conexión pooled SQLite</i>
------------------	--------------------------------------

Description

cie10_sql mantiene una única conexión SQLite reutilizable ("pooled") abierta al archivo cie10.db durante la sesión, en lugar de abrir y cerrar una conexión por cada consulta. Mientras esa conexión está abierta, SQLite mantiene un "lock" (bloqueo) sobre el archivo .db: es la forma en que SQLite evita lecturas/escrituras concurrentes inconsistentes sobre el mismo archivo. Esta función cierra esa conexión y libera el lock.

Solo hace falta llamarla manualmente en dos casos: cuando se va a eliminar o reemplazar el archivo cie10.db por fuera del paquete (por ejemplo, con herramientas del sistema operativo), o al finalizar un proceso batch largo para no dejar el archivo bloqueado. Para usar `cie10_clear_cache()` no es necesario llamarla antes: esa función ya cierra la conexión pooled internamente. Si no se libera el lock, el archivo .db puede seguir abierto hasta que termine la sesión de R; en la práctica esto rara vez es un problema porque cada sesión de R tiene su propia conexión, pero impide que otro proceso externo (no R) edite el archivo mientras la conexión esté abierta.

Usage

```
cie10_disconnect()
```

Value

Sin valor de retorno, se llama por sus efectos secundarios (cierra la conexión SQLite pooled).

See Also

`cie10_sql()`, `cie10_clear_cache()`

Other sql_backend: `cie10_clear_cache()`, `cie10_sql()`

Examples

```
# No hay un ejemplo no interactivo: el objeto de conexión vive en un
# entorno interno del paquete (.cie10_env) y no es parte de la API
# publica, por lo que no hay nada que inspeccionar desde afuera.
```

```
cie10_disconnect()
```

cie10_sql

Ejecutar consultas SQL sobre CIE-10 Chile

Description

Permite ejecutar sentencias SQL de solo lectura sobre la tabla cie10, el mismo dataset que entrega `cie10_cl`. Útil para consultas que no están cubiertas por `cie_search()/cie_lookup()` (agregaciones, conteos por capítulo, joins con datos propios cargados en la misma conexión, etc.). Para aprender SQL desde cero puede revisar <https://www.w3schools.com/sql/> o la documentación de SQLite (https://www.sqlite.org/lang_select.html).

Usage

```
cie10_sql(query, close = lifecycle::deprecated())
```

Arguments

query	String SQL válido SQLite. Soporta SELECT, WHERE, JOIN, FROM, ORDER BY, GROUP BY y HAVING. Por seguridad solo se permiten sentencias SELECT (sin escritura ni múltiples sentencias).
close	[Deprecated] Ignorado - la conexión es pooled y se gestiona automáticamente. Será eliminado en una versión futura.

Value

tibble con el resultado de la consulta

See Also

[cie10_cl](#), [cie10_clear_cache\(\)](#), [cie10_disconnect\(\)](#), [cie_search\(\)](#), [cie_guide\(\)](#)
Other sql_backend: [cie10_clear_cache\(\)](#), [cie10_disconnect\(\)](#)

Examples

```
# Buscar diabetes
cie10_sql("SELECT codigo, descripcion FROM cie10 WHERE codigo LIKE 'E11%')

# Contar por capitulo
cie10_sql("SELECT capitulo, COUNT(*) n FROM cie10 GROUP BY capitulo")
```

cie11_search	<i>Buscar códigos CIE-11 vía API OMS</i>
--------------	--

Description

Buscar códigos CIE-11 vía API OMS

Usage

```
cie11_search(
  text,
  api_key = get_icd_api_key(),
  lang = c("es", "en"),
  max_results = 10,
  release = "2024-01",
  texto = lifecycle::deprecated()
)
```

Arguments

text	String término búsqueda español/inglés
api_key	String opcional, Client ID + Secret OMS separados ":". Por defecto se lee desde la variable de entorno ICD_API_KEY vía <code>get_icd_api_key()</code> . Obtener credenciales en: https://icd.who.int/icdapi
lang	Character, idioma respuesta ("es" o "en")
max_results	Integer, máximo resultados (default 10)
release	Character, versión de release CIE-11 a consultar (default "2024-01"). Ver releases disponibles en la API OMS.
texto	[Deprecated] Use text.

Details**Seguridad de la API key:**

NUNCA escribas la llave literal en tus scripts (p. ej. `api_key = "tu_id:tu_secret"`): si compartes el código (repositorios, correos, capturas) expondrías tus credenciales de forma accidental. La vía recomendada es guardar la llave en la variable de entorno ICD_API_KEY, por ejemplo editando `~/Renviron` con `usethis::edit_r_environ()`; `get_icd_api_key()` la lee automáticamente.

El argumento `api_key` existe solo para circunstancias excepcionales (p. ej. manejar múltiples llaves, o entornos donde no es posible fijar una variable de entorno).

Value

tibble con códigos CIE-11 + títulos o vacío si error

See Also

`cie_search()`, `cie_lookup()`, `cie_guide()`

Other `api_who`: `get_icd_api_key()`

Examples

```
# Ver parámetros disponibles
args(cie11_search)

# Requiere credenciales OMS gratuitas (https://icd.who.int/icdapi)
cie11_search("depresion mayor")

# Causas frecuentes de egreso en Chile
cie11_search("diabetes mellitus")
cie11_search("hipertension esencial")
```

`cie_comorbid`*Calcular comorbilidades Charlson/Elixhauser para Chile*

Description

Calcular comorbilidades Charlson/Elixhauser para Chile

Usage

```
cie_comorbid(data, id, code, map = c("charlson", "elixhauser"), assign0 = TRUE)
```

Arguments

<code>data</code>	data.frame con columnas id paciente + codigos CIE-10
<code>id</code>	String nombre columna identificador paciente
<code>code</code>	String nombre columna con codigos CIE-10 (uno por fila)
<code>map</code>	Character, esquema comorbilidad ("charlson" o "elixhauser")
<code>assign0</code>	Logical, asignar 0 si sin comorbilidad (default TRUE)

Value

tibble ancho con scores comorbilidad por paciente

See Also

[cie_map_comorbid\(\)](#), [cie_norm\(\)](#)

Other comorbidities: [cie_map_comorbid\(\)](#)

Examples

```
# Ver documentacion de parametros
args(cie_comorbid)

df <- data.frame(
  id_pac = c(1, 1, 2, 2),
  diag = c("E11.0", "I21.0", "C50.9", "E10.9")
)
cie_comorbid(df, id = "id_pac", code = "diag", map = "charlson")
```

cie_describe	<i>Obtener descripción de códigos CIE-10 (vector)</i>
--------------	---

Description

Devuelve un vector character con la descripción de cada código, pensado para usar dentro de `dplyr::mutate()` sin necesidad de un `left_join` contra `cie10_cl`.

Usage

```
cie_describe(
  codes,
  normalize = FALSE,
  default = NA_character_,
  codigos = lifecycle::deprecated()
)
```

Arguments

<code>codes</code>	Character vector de códigos CIE-10 (ej. "E11.0", <code>c("E11.0", "I10")</code>).
<code>normalize</code>	Logical, intentar normalizar los códigos antes de buscar la descripción? (default FALSE). Usar TRUE para limpiar formatos (ej. "E110" -> "E11.0"); usar FALSE para auditar la calidad original del registro.
<code>default</code>	Valor devuelto cuando un código no se encuentra en el catálogo. Default <code>NA_character_</code> .
<code>codigos</code>	[Deprecated] Use <code>codes</code> .

Value

Character vector del mismo largo que `codes` con la descripción oficial MINSAL/DEIS. `NA_character_` (o `default`) para códigos sin match.

See Also

[cie_lookup\(\)](#) para resultado como tibble con todas las columnas; [cie_norm\(\)](#) para normalización.
Other search: [cie_guia_busqueda\(\)](#), [cie_guide\(\)](#), [cie_lookup\(\)](#), [cie_search\(\)](#)

Examples

```
# Auditoría: buscar tal cual (E110 no existe sin punto)
cie_describe("E110", normalize = FALSE)

# Rescate: normalizar antes de buscar
cie_describe("E110", normalize = TRUE)

# Uso típico en auditoría VIU (contar fallos de origen)
diags <- c("E11.0", "E110", "I10X", "INVALIDO")
```

```
descripciones <- cie_describe(diags, normalize = FALSE)
sum(is.na(descripciones)) # Detecta 3 errores de registro
```

cie_expand	<i>Expandir código jerárquico (ej. E11 -> E11.0-E11.9)</i>
------------	---

Description

Expandir código jerárquico (ej. E11 -> E11.0-E11.9)

Usage

```
cie_expand(code, codigo = lifecycle::deprecated())
```

Arguments

code	String código padre (ej. "E11")
codigo	[Deprecated] Use code.

Value

Character vector con todos los códigos hijos del código padre. Vector vacío si el código no existe en la base de datos.

See Also

[cie_norm\(\)](#), [cie_lookup\(\)](#), [cie_guide\(\)](#)

Other validacion: [cie_norm\(\)](#), [cie_normalizar\(\)](#), [cie_validate_vector\(\)](#)

Examples

```
cie_expand("E11")
```

cie_guide	<i>Guía de funciones de búsqueda CIE-10</i>
-----------	---

Description

Muestra tabla comparativa de cuándo usar cada función de búsqueda.

Usage

```
cie_guide()
```

Value

tibble con guía comparativa de funciones de búsqueda

See Also

[cie_search\(\)](#), [cie_lookup\(\)](#), [cie_short\(\)](#)

Other search: [cie_describe\(\)](#), [cie_guia_busqueda\(\)](#), [cie_lookup\(\)](#), [cie_search\(\)](#)

Examples

```
cie_guide()
```

cie_lookup

Búsqueda exacta por código CIE-10

Description

Búsqueda exacta por código CIE-10

Usage

```
cie_lookup(
  code,
  expand = FALSE,
  normalize = TRUE,
  full_description = FALSE,
  extract = FALSE,
  check_siglas = FALSE,
  include_uso_cl = TRUE,
  only_uso_cl = FALSE,
  codigo = lifecycle::deprecated(),
  expandir = lifecycle::deprecated(),
  normalizar = lifecycle::deprecated(),
  descripcion_completa = lifecycle::deprecated()
)
```

Arguments

code	Character vector de códigos (ej. "E11", "E11.0", c("E11.0", "Z00")) o rango (ej. "E10-E14"). Acepta vectores. Soporta formatos: con punto (E11.0), sin punto (E110), o solo categoría (E11).
expand	Logical, expandir jerarquía completa (default FALSE)
normalize	Logical, normalizar formato de códigos automáticamente (default TRUE)
full_description	Logical, agregar columna descripcion_completa con formato "CODIGO - DESCRIPCION" (default FALSE)

extract	Logical, extraer código CIE-10 de texto con prefijos/sufijos (default FALSE). IMPORTANTE: Solo usar con código ESCALAR (longitud 1). Ejemplo: "CIE:E11.0" -> "E11.0", "E11.0-confirmado" -> "E11.0". Para vectores múltiples usar extract=FALSE (default).
check_siglas	Logical, buscar siglas médicas comunes (default FALSE). Ejemplo: "IAM" -> I21.0 (Infarto agudo miocardio)
include_uso_cl	Logical, incluir columna uso_cl en el output (default TRUE). El default difiere de <code>cie_search()</code> (FALSE) para preservar el contrato histórico de cada función. Valores posibles: "principal", "legado", "causa_externa", "etiologico", "causa_externa principal".
only_uso_cl	Logical, filtrar a códigos vigentes de uso clínico en Chile (default FALSE). Cuando es TRUE, excluye los códigos con uso_cl == "legado" (es decir, conserva principal, causa_externa, etiologico y sus combinaciones).
codigo	[Deprecated] Use code.
expandir	[Deprecated] Use expand.
normalizar	[Deprecated] Use normalize.
descripcion_completa	[Deprecated] Use full_description.

Value

tibble con codigo(s) matcheado(s)

See Also

[cie_search\(\)](#), [cie_norm\(\)](#), [cie_expand\(\)](#), [cie_guide\(\)](#)

Other search: [cie_describe\(\)](#), [cie_guia_búsqueda\(\)](#), [cie_guide\(\)](#), [cie_search\(\)](#)

Examples

```
# Búsqueda directa por código
cie_lookup("E11.0")

cie_lookup("E110") # Sin punto
cie_lookup("E11") # Solo categoría
cie_lookup("E11", expand = TRUE) # Todos E11.x
# Vectorizado - múltiples códigos y formatos
cie_lookup(c("E11.0", "Z00", "I10"))
# Con descripción completa
cie_lookup("E110", full_description = TRUE)
# Extraer código de texto con ruido (solo código escalar)
cie_lookup("CIE:E11.0", extract = TRUE)
cie_lookup("E11.0-confirmado", extract = TRUE)
# Buscar por siglas médicas
cie_lookup("IAM", check_siglas = TRUE)
cie_lookup("DM2", check_siglas = TRUE)
# Filtrar a códigos vigentes de uso clínico Chile (excluye 'legado')
```

```
cie_lookup("E11", expand = TRUE, only_uso_cl = TRUE)
# Omitir columna uso_cl en el output
cie_lookup("E11.0", include_uso_cl = FALSE)
```

cie_map_comorbid

Mapeo manual de grupos de comorbilidad específicos de Chile

Description

Agrupar códigos CIE-10 chilenos en categorías de comorbilidad MINSAL. Basado en Decreto 1301/2016 MINSAL + icd::icd10_map_charlson.

Usage

```
cie_map_comorbid(codes, codigos = lifecycle::deprecated())
```

Arguments

codes	Character vector de codigos
codigos	[Deprecated] Use codes.

Value

tibble con columnas: codigo, categoria

See Also

[cie_comorbid\(\)](#), [cie_norm\(\)](#)

Other comorbidities: [cie_comorbid\(\)](#)

Examples

```
cie_map_comorbid(c("E11.0", "I50.9", "C50.9"))
```

cie_norm

*Normalizar códigos CIE-10 a formato con punto***Description**

Convierte códigos CIE-10 de diferentes formatos al formato estándar (con punto). Maneja múltiples variaciones de entrada comunes en datos clínicos.

Usage

```

cie_norm(
  codes,
  search_db = TRUE,
  codigos = lifecycle::deprecated(),
  buscar_db = lifecycle::deprecated()
)

cie_normalize(
  codes,
  search_db = TRUE,
  codigos = lifecycle::deprecated(),
  buscar_db = lifecycle::deprecated()
)

```

Arguments

codes	Character vector de códigos en cualquier formato
search_db	Logical, buscar código en base de datos si no se encuentra exacto (default TRUE)
codigos	[Deprecated] Use codes.
buscar_db	[Deprecated] Use search_db.

Details

La normalización incluye:

- Conversión a mayúsculas
- Eliminación de espacios (inicio, fin e internos)
- Eliminación de símbolos daga y asterisco (codificación dual)
- Conversión de guiones a puntos (I10-0 -> I10.0)
- Eliminación de puntos iniciales (.I10 -> I10)
- Corrección de puntos múltiples (E..11 -> E.11)
- Eliminación de sufijo X en códigos cortos, incluido el punto previo si lo hay (I10X -> I10, E11.X -> E11)
- Preservación de X en códigos largos (placeholder 7o carácter)

- Agregado de punto en posición correcta (E110 -> E11.0)

El sistema de daga/asterisco indica codificación dual donde la daga marca la enfermedad subyacente y el asterisco la manifestación. Ambos símbolos se eliminan para normalización.

Value

Character vector con códigos normalizados al formato con punto

See Also

[cie_validate_vector\(\)](#), [cie_expand\(\)](#), [cie_lookup\(\)](#), [cie_guide\(\)](#)

Other validation: [cie_expand\(\)](#), [cie_normalizar\(\)](#), [cie_validate_vector\(\)](#)

Examples

```
cie_norm("E110")      # Retorna "E11.0"
cie_norm("E11")       # Retorna "E11" (categoría)
cie_norm("I10X")      # Retorna "I10" (elimina X)
cie_norm("E 11 0")    # Retorna "E11.0" (espacios internos)
cie_norm("I10-0")     # Retorna "I10.0" (guion a punto)
cie_norm(paste0("A17.0", intToUtf8(0x2020))) # "A17.0" (elimina daga)
cie_norm("G01*")       # "G01" (elimina asterisco)
cie_norm(c("E110", "I10X", "Z00")) # Vectorizado
```

cie_search

Búsqueda difusa (fuzzy) de términos médicos CIE-10

Description

Busca en descripciones CIE-10 usando múltiples estrategias:

1. Expansión de siglas médicas (IAM, TBC, DM, etc.)
2. Búsqueda exacta por subcadena (más rápida)
3. Búsqueda fuzzy con Jaro-Winkler (tolera typos)

Usage

```
cie_search(
  text,
  threshold = 0.7,
  max_results = 50,
  field = c("descripcion", "inclusion"),
  only_fuzzy = FALSE,
  verbose = TRUE,
  include_uso_cl = FALSE,
  only_uso_cl = FALSE,
  texto = lifecycle::deprecated(),
```

```

    campo = lifecycle::deprecated(),
    solo_fuzzy = lifecycle::deprecated()
)

```

Arguments

text	String término médico en español o sigla (ej. "diabetes", "IAM", "TBC")
threshold	Numeric entre 0 y 1, umbral similitud Jaro-Winkler (default 0.70)
max_results	Integer, máximo resultados a retornar (default 50)
field	Character, campo búsqueda ("descripcion" o "inclusion")
only_fuzzy	Logical, usar solo búsqueda fuzzy sin búsqueda exacta (default FALSE)
verbose	Logical, mostrar mensajes informativos (default TRUE). Usar FALSE en scripts.
include_uso_cl	Logical, incluir columna uso_cl en el output (default FALSE). El default difiere de cie_lookup() (TRUE) para preservar el contrato histórico de cada función. Valores posibles: "principal", "legado", "causa_externa", "etiologico", "causa_externa principal".
only_uso_cl	Logical, filtrar a códigos vigentes de uso clínico en Chile (default FALSE). Cuando es TRUE, excluye los códigos con uso_cl == "legado".
texto	[Deprecated] Use text.
campo	[Deprecated] Use field.
solo_fuzzy	[Deprecated] Use only_fuzzy.

Details

La búsqueda es tolerante a tildes: "neumonia" (sin tilde) encuentra "neumonía" (con tilde) en el catálogo. Soporta siglas médicas comunes: "IAM" busca "infarto agudo miocardio".

Value

tibble ordenado por score descendente (1.0 = coincidencia exacta). Si el text corresponde a una sigla médica, se expande automáticamente antes de buscar.

See Also

[cie_lookup\(\)](#), [cie_short\(\)](#), [cie10_sql\(\)](#)

Other search: [cie_describe\(\)](#), [cie_guia_busqueda\(\)](#), [cie_guide\(\)](#), [cie_lookup\(\)](#)

Examples

```

# Búsqueda básica
cie_search("diabetes")

cie_search("neumonia")

# Búsqueda por siglas médicas
cie_search("IAM")

```

```

cie_search("DM2")

# Tolerante a tildes y typos
cie_search("diabetis")

# Buscar en inclusiones
cie_search("bacteriana", field = "inclusion")
# Filtrar a códigos vigentes Chile (excluye 'legado')
cie_search("diabetes", only_uso_cl = TRUE)
# Mostrar la columna uso_cl en el output
cie_search("diabetes", include_uso_cl = TRUE)

```

cie_short

Listar siglas médicas soportadas

Description

Muestra todas las siglas médicas que pueden usarse en [cie_search\(\)](#). "Sigla" se conserva como término local (concepto médico chileno) en la columna de salida; el nombre de la función usa `cie_short` por consistencia con el ecosistema R (verbos cortos en inglés).

Usage

```
cie_short(category = NULL, categoria = lifecycle::deprecated())
```

Arguments

<code>category</code>	Character opcional, filtrar por categoría. Los valores (sin tildes, son tokens usados también para matching interno) son: "cardiovascular", "respiratoria", "metabolica", "gastrointestinal", "infecciosa", "oncologica", "reumatologica", "neurologica", "psiquiatrica", "traumatologica", "pediatrica", "gineco_obstetrica". Si es NULL (default), retorna todas las siglas.
<code>categoria</code>	[Deprecated] Use category.

Value

tibble con columnas: sigla, termino_busqueda, categoria

See Also

[cie_search\(\)](#), [cie_lookup\(\)](#)
 Other busqueda: [cie_siglas\(\)](#)

Examples

```
# Ver todas las siglas
cie_short()

# Filtrar por categoría
cie_short("cardiovascular")
cie_short("oncologica")

# Buscar una sigla específica
cie_short() |> dplyr::filter(sigla == "iam")
```

cie_table

Generar tabla HTML interactiva GT de código CIE-10

Description

Muestra la jerarquía de un código CIE-10 (categoría + subcategorías) como una tabla gt. Las columnas "Incluye" y "Excluye" pueden aparecer vacías en subcategorías: el catálogo MINSAL/DEIS no puebla esos campos en todos los niveles (suelen estar solo en la categoría de 3 dígitos). Para evitar confusión visual, los NA se reemplazan por un guion largo (em dash).

Usage

```
cie_table(code, codigo = lifecycle::deprecated())
```

Arguments

code	String código de longitud 1, un solo código (ej. "E11" muestra la jerarquía).
codigo	[Deprecated] Use code.

Value

Objeto de clase gt_tbl (tabla HTML interactiva).

See Also

[cie_search\(\)](#), [cie_lookup\(\)](#), [cie_guide\(\)](#)

Examples

```
cie_table("E11") # Diabetes mellitus tipo 2 completo
```

cie_validate_vector	Validar vector de códigos CIE-10 formato
---------------------	--

Description

Validar vector de códigos CIE-10 formato

Usage

```
cie_validate_vector(codes, strict = FALSE, codigos = lifecycle::deprecated())
```

Arguments

codes	Character vector códigos (ej. c("E11.0", "Z00.0"))
strict	Logical, validar existencia en DB (default FALSE)
codigos	[Deprecated] Use codes.

Value

Logical vector de la misma longitud que codes. TRUE si el código tiene formato CIE-10 válido (y existe en DB si strict = TRUE).

See Also

[cie_norm\(\)](#), [cie_expand\(\)](#)

Other validacion: [cie_expand\(\)](#), [cie_norm\(\)](#), [cie_normalizar\(\)](#)

Examples

```
cie_validate_vector(c("E11.0", "INVALIDO", "Z00"))
```

get_icd_api_key	Obtener la API key de la OMS desde el entorno
-----------------	---

Description

Lee la variable de entorno ICD_API_KEY y aborta con un error informativo si no está configurada. Es el valor por defecto del argumento api_key de [cie11_search\(\)](#), siguiendo el patrón recomendado por httr2 para envolver APIs.

Usage

```
get_icd_api_key()
```

Details

Configura la variable editando ~/.Renviron con `usethis::edit_r_environ()`:

```
ICD_API_KEY=tu_client_id:tu_client_secret
```

Nunca escribas la llave literal en scripts que vayas a compartir.

Value

String con la API key en formato "client_id:client_secret".

See Also

[cie11_search\(\)](#)

Other api_who: [cie11_search\(\)](#)

Examples

```
# Requiere ICD_API_KEY configurada (ver Details)
try(get_icd_api_key())
```

Index

- * **api_who**
 - cie11_search, 6
 - get_icd_api_key, 19
- * **busqueda**
 - cie_short, 17
- * **comorbidities**
 - cie_comorbid, 8
 - cie_map_comorbid, 13
- * **datasets**
 - cie10_cl, 2
- * **search**
 - cie_describe, 9
 - cie_guide, 10
 - cie_lookup, 11
 - cie_search, 15
- * **sql_backend**
 - cie10_clear_cache, 4
 - cie10_disconnect, 4
 - cie10_sql, 5
- * **validacion**
 - cie_expand, 10
 - cie_norm, 14
 - cie_validate_vector, 19
- * **visualization**
 - cie_table, 18

cie10_cl, 2, 4–6

cie10_clear_cache, 4

cie10_clear_cache(), 5, 6

cie10_disconnect, 4

cie10_disconnect(), 4, 6

cie10_sql, 5

cie10_sql(), 3–5, 16

cie11_search, 6

cie11_search(), 19, 20

cie_comorbid, 8

cie_comorbid(), 13

cie_describe, 9

cie_describe(), 11, 12, 16

cie_expand, 10

cie_expand(), 12, 15, 19

cie_guia_busqueda(), 9, 11, 12, 16

cie_guide, 10

cie_guide(), 6, 7, 9, 10, 12, 15, 16, 18

cie_lookup, 11

cie_lookup(), 3–5, 7, 9–11, 15–18

cie_map_comorbid, 13

cie_map_comorbid(), 8

cie_norm, 14

cie_norm(), 8–10, 12, 13, 19

cie_normalizar(), 10, 15, 19

cie_normalize (cie_norm), 14

cie_search, 15

cie_search(), 3–7, 9, 11, 12, 17, 18

cie_short, 17

cie_short(), 11, 16

cie_siglas(), 17

cie_table, 18

cie_validate_vector, 19

cie_validate_vector(), 10, 15

get_icd_api_key, 19

get_icd_api_key(), 7