Package ‘comexr’

March 17, 2026

Title Client for the Brazilian Foreign Trade Statistics API ('ComexStat')

Version 0.2.1

Date 2026-03-10

Description Interface to the 'ComexStat' API <https://comexstat.mdic.gov.br/> from the Brazilian Ministry of Development, Industry, Trade and Services (MDIC). Provides access to detailed export and import data, including general trade statistics (1997-present), city-level data, historical data (1989-1996), and auxiliary tables with product codes (NCM - Nomenclatura Comum do Mercosul, NBM - Nomenclatura Brasileira de Mercadorias, HS - Harmonized System), countries, economic classifications (CGCE - Classificacao por Grandes Categorias Economicas, SITC - Standard International Trade Classification, ISIC - International Standard Industrial Classification), and other categories. Uses only 'httr2' for HTTP requests and 'cli' for console messages.

License MIT + file LICENSE

URL https://strategicprojects.github.io/comexr/, https://github.com/StrategicProjects/comexr

BugReports https://github.com/StrategicProjects/comexr/issues

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports httr2 (>= 1.0.0), cli (>= 3.6.0)

Suggests testthat (>= 3.0.0), knitr, rmarkdown, tibble

VignetteBuilder knitr

Config/testthat/edition 3

Language en-US

NeedsCompilation no

Author Andre Leite [aut, cre],
Marcos Wasilew [aut],
Hugo Vasconcelos [aut],
Carlos Amorin [aut],
Diogo Bezerra [aut]

Maintainer Andre Leite <leite@castlab.org>

Repository CRAN

Date/Publication 2026-03-17 14:40:03 UTC

Contents

`comex_available_years` *Get available years for queries*

Description

Returns the first and last years available for queries in the API.

Usage

```
comex_available_years(type = "general", verbose = FALSE)
```

Arguments

`type`	Data type: `"general"`, `"city"`, or `"historical"`. Default: `"general"`.
`verbose`	Logical. Show progress messages. Default: `FALSE`.

Value

A list with `min` and `max` year values.

Examples

```
## Not run:
comex_available_years()
comex_available_years("city")
comex_available_years("historical")

## End(Not run)
```

`comex_blocs` *Get economic blocs table*

Description

Returns the economic blocs table with codes and names. Economic blocs represent trade agreements between countries and regions.

Usage

```
comex_blocs(language = "en", search = NULL, add = NULL, verbose = FALSE)
```

Arguments

`language`	Language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`search`	Optional search term to filter results.
`add`	Optional related table to include (e.g. `"country"`).
`verbose`	Logical. Default: `FALSE`.

Value

A data.frame with economic bloc codes and names.

Examples

```
## Not run:
comex_blocs()
comex_blocs(search = "mercosul")
comex_blocs(add = "country")

## End(Not run)
```

---

comex_cgce *Get CGCE (Classification by Broad Economic Categories) table*

Description

Returns the CGCE classification table from the `/tables/classifications` endpoint. CGCE groups products by use or economic purpose (e.g. capital goods, intermediate goods, consumer goods).

Usage

```
comex_cgce(
  language = "en",
  search = NULL,
  add = NULL,
  page = NULL,
  per_page = NULL,
  verbose = FALSE
)
```

Arguments

`language`	Language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`search`	Optional search term to filter results.
`add`	Optional related table to include (e.g. `"ncm"`).
`page`	Page number for pagination. Default: `NULL` (all results).
`per_page`	Number of results per page. Default: `NULL`.
`verbose`	Logical. Show progress messages. Default: `FALSE`.

Value

A data.frame with CGCE codes and descriptions.

Examples

```
## Not run:
# All CGCE classifications
comex_cgce()

# Search within CGCE
comex_cgce(search = "110")

## End(Not run)
```

---

comex_cities *Get Brazilian cities table*

Description

Returns the Brazilian cities table with codes and names.

Usage

```
comex_cities(verbose = FALSE)
```

Arguments

verbose	Logical. Default: FALSE.

Value

A data.frame with city (IBGE) codes and names.

Examples

```
## Not run:
comex_cities()

## End(Not run)
```

`comex_city_detail` *Get city details*

Description

Returns details for a specific Brazilian city.

Usage

```
comex_city_detail(city_id, verbose = FALSE)
```

Arguments

`city_id`	IBGE city code (e.g. `5300050`).
`verbose`	Logical. Default: `FALSE`.

Value

A list with city details.

Examples

```
## Not run:
comex_city_detail(5300050)

## End(Not run)
```

`comex_countries` *Get countries table*

Description

Returns the countries table with codes and names.

Usage

```
comex_countries(search = NULL, verbose = FALSE)
```

Arguments

`search`	Optional search term to filter results (e.g. `"br"`).
`verbose`	Logical. Show progress messages. Default: `FALSE`.

Value

A data.frame with country codes and names.

Examples

```
## Not run:
comex_countries()
comex_countries(search = "bra")

## End(Not run)
```

`comex_country_detail` *Get country details*

Description

Returns details for a specific country by its code.

Usage

```
comex_country_detail(id, verbose = FALSE)
```

Arguments

`id`	Country code (e.g. `105` for Brazil).
`verbose`	Logical. Default: `FALSE`.

Value

A list with country details.

Examples

```
## Not run:
comex_country_detail(105)

## End(Not run)
```

`comex_customs_units` *Get customs units (URF) table*

Description

Returns the customs units table (Unidades da Receita Federal) with codes and names. These are the Federal Revenue Service administrative units responsible for overseeing foreign trade operations.

Usage

```
comex_customs_units(verbose = FALSE)
```

Arguments

`verbose`	Logical. Default: `FALSE`.

Value

A data.frame with customs unit codes and names.

Examples

```
## Not run:
comex_customs_units()

## End(Not run)
```

`comex_customs_unit_detail` *Get customs unit details*

Description

Returns details for a specific customs unit (URF).

Usage

```
comex_customs_unit_detail(urf_id, verbose = FALSE)
```

Arguments

`urf_id`	Customs unit code (e.g. `8110000`).
`verbose`	Logical. Default: `FALSE`.

Value

A list with customs unit details.

Examples

```
## Not run:
comex_customs_unit_detail(8110000)

## End(Not run)
```

`comex_details` *Get available detail/grouping fields*

Description

Returns the list of detail fields that can be used to group query results.

Usage

```
comex_details(type = "general", language = "en", verbose = FALSE)
```

Arguments

`type`	Data type: `"general"`, `"city"`, or `"historical"`.
`language`	Language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`verbose`	Logical. Show progress messages. Default: `FALSE`.

Value

A data.frame with available details.

Examples

```
## Not run:
comex_details()
comex_details("city")
comex_details("historical")

## End(Not run)
```

comex_export *Query exports*

Description

Shortcut for `comex_query()` with `flow = "export"`.

Usage

```
comex_export(
  start_period,
  end_period,
  details = NULL,
  filters = NULL,
  month_detail = TRUE,
  metric_fob = TRUE,
  metric_kg = TRUE,
  metric_statistic = FALSE,
  metric_freight = FALSE,
  metric_insurance = FALSE,
  metric_cif = FALSE,
  language = "en",
  verbose = TRUE
)
```

Arguments

`start_period`	Start period in `"YYYY-MM"` format (e.g. `"2023-01"`).
`end_period`	End period in `"YYYY-MM"` format (e.g. `"2023-12"`).
`details`	Character vector of detail/grouping fields. Options: **Geographic:** `"country"`, `"bloc"`, `"state"`, `"city"`, `"transport_mode"`, `"customs_unit"` **Products:** `"ncm"`, `"hs6"` (or `"sh6"`), `"hs4"` (or `"sh4"`), `"hs2"` (or `"sh2"`), `"section"` **CGCE:** `"cgce_n1"`, `"cgce_n2"`, `"cgce_n3"` **SITC/CUCI:** `"sitc_section"`, `"sitc_chapter"`, `"sitc_position"`, `"sitc_subposition"`, `"sitc_item"` **ISIC:** `"isic_section"`, `"isic_division"`, `"isic_group"`, `"isic_class"` **Other:** `"company_size"` (imports only)
`filters`	Named list of filters. Names should match detail field names. Example: `list(country = c(160, 249), state = c(26, 13))`
`month_detail`	Logical. If `TRUE`, break down results by month. Default: `FALSE`.
`metric_fob`	Logical. Include FOB value (US$). Default: `TRUE`.
`metric_kg`	Logical. Include net weight (kg). Default: `TRUE`.

`metric_statistic`	Logical. Include statistical quantity. Default: `FALSE`.
`metric_freight`	Logical. Include freight value (US$, imports only). Default: `FALSE`.
`metric_insurance`	Logical. Include insurance value (US$, imports only). Default: `FALSE`.
`metric_cif`	Logical. Include CIF value (US$, imports only). Default: `FALSE`.
`language`	Response language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`verbose`	Logical. Show progress messages. Default: `TRUE`.

Value

A data.frame (or tibble) with export data.

Examples

```
## Not run:
comex_export(
  start_period = "2023-01",
  end_period = "2023-12",
  details = "country"
)

## End(Not run)
```

`comex_filters` *Get available filters*

Description

Returns the list of filter types available for API queries.

Usage

```
comex_filters(type = "general", language = "en", verbose = FALSE)
```

Arguments

`type`	Data type: `"general"`, `"city"`, or `"historical"`.
`language`	Language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`verbose`	Logical. Show progress messages. Default: `FALSE`.

Value

A data.frame with available filters.

Examples

```
## Not run:
comex_filters()
comex_filters("city")
comex_filters("historical")

## End(Not run)
```

`comex_filter_values` *Get values for a specific filter*

Description

Returns the possible values for a given filter name.

Usage

```
comex_filter_values(filter, type = "general", language = "en", verbose = FALSE)
```

Arguments

`filter`	Filter name as returned by `comex_filters()` (e.g. `"country"`, `"state"`, `"ncm"`, `"economicBlock"`).
`type`	Data type: `"general"`, `"city"`, or `"historical"`.
`language`	Language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`verbose`	Logical. Show progress messages. Default: `FALSE`.

Value

A data.frame with filter values.

Examples

```
## Not run:
comex_filter_values("country")
comex_filter_values("state", type = "city")
comex_filter_values("economicBlock")

## End(Not run)
```

comex_historical *Query historical foreign trade data (1989-1996)*

Description

Query the historical data endpoint of the ComexStat API to retrieve Brazilian export and import data from 1989 to 1996, before the SISCOMEX system was implemented. Historical data uses the NBM (Brazilian Nomenclature of Goods) classification.

Usage

```
comex_historical(
  flow = "export",
  start_period,
  end_period,
  details = NULL,
  filters = NULL,
  month_detail = TRUE,
  metric_fob = TRUE,
  metric_kg = TRUE,
  language = "en",
  verbose = TRUE
)
```

Arguments

`flow`	Trade flow: `"export"` or `"import"`.
`start_period`	Start period in `"YYYY-MM"` format (e.g. `"1990-01"`).
`end_period`	End period in `"YYYY-MM"` format (e.g. `"1996-12"`).
`details`	Character vector of detail/grouping fields. Options: `"country"`, `"state"`, `"nbm"`.
`filters`	Named list of filters.
`month_detail`	Logical. If `TRUE`, break down by month. Default: `TRUE`.
`metric_fob`	Logical. Include FOB value (US$). Default: `TRUE`.
`metric_kg`	Logical. Include net weight (kg). Default: `TRUE`.
`language`	Response language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`verbose`	Logical. Show progress messages. Default: `TRUE`.

Details

Historical data differs from general data:

- Available period: **1989 to 1996** only
- Limited details: `"country"`, `"state"`, `"nbm"`
- Product classification is **NBM** (not NCM)
- Only **FOB and KG** metrics are available (no statistic, freight, insurance, or CIF)

Value

A data.frame (or tibble) with query results.

Examples

```
## Not run:
# Historical exports 1995-1996 by country
comex_historical(
  flow = "export",
  start_period = "1995-01",
  end_period = "1996-12",
  details = "country"
)

## End(Not run)
```

`comex_hs` *Get Harmonized System (HS) tables*

Description

Returns Harmonized System classification tables. The HS is an international product nomenclature developed by the World Customs Organization (WCO).

Usage

```
comex_hs(
  language = "en",
  add = NULL,
  page = NULL,
  per_page = NULL,
  verbose = FALSE
)
```

Arguments

`language`	Language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`add`	Optional related table to include (e.g. `"ncm"`).
`page`	Page number for pagination. Default: `NULL`.
`per_page`	Number of results per page. Default: `NULL`.
`verbose`	Logical. Default: `FALSE`.

Details

The Harmonized System is organized hierarchically:

- **Section**: 21 sections (broadest grouping)
- **Chapter (HS2)**: ~97 chapters (2 digits)
- **Heading (HS4)**: 4 digits
- **Subheading (HS6)**: 6 digits (most detailed)

The NCM adds 2 more digits to the HS6 code.

Value

A data.frame with HS codes and descriptions.

Examples

```
## Not run:
# All HS classifications
comex_hs()

# With related NCM codes
comex_hs(add = "ncm", per_page = 10)

## End(Not run)
```

`comex_import` *Query imports*

Description

Shortcut for `comex_query()` with `flow = "import"`.

Usage

```
comex_import(
  start_period,
  end_period,
  details = NULL,
  filters = NULL,
  month_detail = TRUE,
  metric_fob = TRUE,
  metric_kg = TRUE,
  metric_statistic = FALSE,
  metric_freight = FALSE,
  metric_insurance = FALSE,
  metric_cif = FALSE,
  language = "en",
  verbose = TRUE
)
```

Arguments

`start_period`	Start period in `"YYYY-MM"` format (e.g. `"2023-01"`).
`end_period`	End period in `"YYYY-MM"` format (e.g. `"2023-12"`).
`details`	Character vector of detail/grouping fields. Options: **Geographic:** `"country"`, `"bloc"`, `"state"`, `"city"`, `"transport_mode"`, `"customs_unit"` **Products:** `"ncm"`, `"hs6"` (or `"sh6"`), `"hs4"` (or `"sh4"`), `"hs2"` (or `"sh2"`), `"section"` **CGCE:** `"cgce_n1"`, `"cgce_n2"`, `"cgce_n3"` **SITC/CUCI:** `"sitc_section"`, `"sitc_chapter"`, `"sitc_position"`, `"sitc_subposition"`, `"sitc_item"` **ISIC:** `"isic_section"`, `"isic_division"`, `"isic_group"`, `"isic_class"` **Other:** `"company_size"` (imports only)
`filters`	Named list of filters. Names should match detail field names. Example: `list(country = c(160, 249), state = c(26, 13))`
`month_detail`	Logical. If `TRUE`, break down results by month. Default: `FALSE`.
`metric_fob`	Logical. Include FOB value (US$). Default: `TRUE`.
`metric_kg`	Logical. Include net weight (kg). Default: `TRUE`.
`metric_statistic`	Logical. Include statistical quantity. Default: `FALSE`.
`metric_freight`	Logical. Include freight value (US$, imports only). Default: `FALSE`.
`metric_insurance`	Logical. Include insurance value (US$, imports only). Default: `FALSE`.
`metric_cif`	Logical. Include CIF value (US$, imports only). Default: `FALSE`.
`language`	Response language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`verbose`	Logical. Show progress messages. Default: `TRUE`.

Value

A data.frame (or tibble) with import data.

Examples

```
## Not run:
comex_import(
  start_period = "2023-01",
  end_period = "2023-12",
  details = "country",
  metric_cif = TRUE
)

## End(Not run)
```

comex_isic — *Get ISIC (International Standard Industrial Classification) table*

Description

Queries the `/tables/product-categories` endpoint to retrieve ISIC classification data. ISIC is an international classification of economic activities developed by the United Nations.

Usage

```
comex_isic(
  language = "en",
  search = NULL,
  add = NULL,
  page = NULL,
  per_page = NULL,
  verbose = FALSE
)
```

Arguments

`language`	Language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`search`	Optional search term to filter results.
`add`	Optional related table to include (e.g. `"ncm"`).
`page`	Page number for pagination. Default: `NULL`.
`per_page`	Number of results per page. Default: `NULL`.
`verbose`	Logical. Show progress messages. Default: `FALSE`.

Value

A data.frame with classification codes and descriptions.

Note

The OpenAPI specification does not define a dedicated ISIC table endpoint. ISIC codes are available as detail/grouping fields in trade queries (e.g. `"isic_section"`, `"isic_division"`). This convenience function queries `/tables/product-categories`, which may return ISIC data alongside CUCI/SITC classifications. You can also look up ISIC values using `comex_filter_values()` with filter names like `"isicSection"`.

Examples

```
## Not run:
# Browse product categories (includes ISIC)
comex_isic()
```

```
# Alternatively, look up ISIC values via filters:
comex_filter_values("isicSection")

## End(Not run)
```

comex_last_update *Get last data update date*

Description

Returns the date of the last data update in the API.

Usage

```
comex_last_update(type = "general", verbose = FALSE)
```

Arguments

`type`	Data type: `"general"`, `"city"`, or `"historical"`. Default: `"general"`.
`verbose`	Logical. Show progress messages. Default: `FALSE`.

Value

A list with last update information.

Examples

```
## Not run:
comex_last_update()
comex_last_update("city")
comex_last_update("historical")

## End(Not run)
```

comex_metrics *Get available metrics*

Description

Returns the list of metrics (values) available for API queries.

Usage

```
comex_metrics(type = "general", language = "en", verbose = FALSE)
```

Arguments

`type`	Data type: `"general"`, `"city"`, or `"historical"`.
`language`	Language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`verbose`	Logical. Show progress messages. Default: `FALSE`.

Value

A data.frame with available metrics and their descriptions.

Examples

```
## Not run:
comex_metrics()
comex_metrics("city")
comex_metrics("historical")

## End(Not run)
```

`comex_nbm` *Get NBM (Brazilian Nomenclature of Goods) table*

Description

Returns the NBM codes table with descriptions. NBM was the nomenclature used in Brazil before NCM adoption and is used only for historical data (1989-1996).

Usage

```
comex_nbm(
  language = "en",
  search = NULL,
  add = NULL,
  page = NULL,
  per_page = NULL,
  verbose = FALSE
)
```

Arguments

`language`	Language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`search`	Optional search term to filter results.
`add`	Optional related table to include (e.g. `"ncm"`).
`page`	Page number for pagination. Default: `NULL`.
`per_page`	Number of results per page. Default: `NULL`.
`verbose`	Logical. Default: `FALSE`.

Value

A data.frame with NBM codes and descriptions.

Examples

```
## Not run:
comex_nbm()
comex_nbm(search = "encomendas", per_page = 5)
comex_nbm(add = "ncm")

## End(Not run)
```

---

comex_nbm_detail *Get NBM code details*

Description

Returns details for a specific NBM code.

Usage

```
comex_nbm_detail(nbm_code, verbose = FALSE)
```

Arguments

nbm_code	NBM code (e.g. "2924101100").
verbose	Logical. Default: FALSE.

Value

A list with NBM details.

Examples

```
## Not run:
comex_nbm_detail("2924101100")

## End(Not run)
```

`comex_ncm` *Get NCM (Mercosur Common Nomenclature) table*

Description

Returns the NCM codes table with descriptions. NCM is the product classification used by Mercosur countries, based on the Harmonized System (HS) with 8 digits.

Usage

```
comex_ncm(
  language = "en",
  search = NULL,
  add = NULL,
  page = NULL,
  per_page = NULL,
  verbose = FALSE
)
```

Arguments

`language`	Language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`search`	Optional search term to filter results (e.g. `"animal"`).
`add`	Optional related table to include in results. Options: `"sh"`, `"cuci"`, `"cgce"`.
`page`	Page number for pagination. Default: `NULL` (all results).
`per_page`	Number of results per page. Default: `NULL`.
`verbose`	Logical. Show progress messages. Default: `FALSE`.

Value

A data.frame with NCM codes and descriptions.

Examples

```
## Not run:
ncm <- comex_ncm()
comex_ncm(search = "animal", per_page = 10)
comex_ncm(add = "cuci")

## End(Not run)
```

`comex_ncm_detail` *Get NCM code details*

Description

Returns details for a specific NCM code, including product description and its HS classification hierarchy.

Usage

```
comex_ncm_detail(ncm_code, verbose = FALSE)
```

Arguments

`ncm_code`	NCM code (8 digits, as character, e.g. `"02042200"`).
`verbose`	Logical. Default: `FALSE`.

Value

A list with NCM details.

Examples

```
## Not run:
comex_ncm_detail("02042200")

## End(Not run)
```

`comex_query` *Query general foreign trade data*

Description

Query the main ComexStat API endpoint to retrieve Brazilian export and import data. Supports filtering and grouping by multiple classifications such as NCM, Harmonized System, countries, states, etc.

Data is available monthly from 1997 to the most recent complete month.

Usage

```
comex_query(
  flow = "export",
  start_period,
  end_period,
  details = NULL,
  filters = NULL,
  month_detail = TRUE,
  metric_fob = TRUE,
  metric_kg = TRUE,
  metric_statistic = FALSE,
  metric_freight = FALSE,
  metric_insurance = FALSE,
  metric_cif = FALSE,
  language = "en",
  verbose = TRUE
)
```

Arguments

`flow`	Trade flow: `"export"` or `"import"`.
`start_period`	Start period in `"YYYY-MM"` format (e.g. `"2023-01"`).
`end_period`	End period in `"YYYY-MM"` format (e.g. `"2023-12"`).
`details`	Character vector of detail/grouping fields. Options: **Geographic:** `"country"`, `"bloc"`, `"state"`, `"city"`, `"transport_mode"`, `"customs_unit"` **Products:** `"ncm"`, `"hs6"` (or `"sh6"`), `"hs4"` (or `"sh4"`), `"hs2"` (or `"sh2"`), `"section"` **CGCE:** `"cgce_n1"`, `"cgce_n2"`, `"cgce_n3"` **SITC/CUCI:** `"sitc_section"`, `"sitc_chapter"`, `"sitc_position"`, `"sitc_subposition"`, `"sitc_item"` **ISIC:** `"isic_section"`, `"isic_division"`, `"isic_group"`, `"isic_class"` **Other:** `"company_size"` (imports only)
`filters`	Named list of filters. Names should match detail field names. Example: `list(country = c(160, 249), state = c(26, 13))`
`month_detail`	Logical. If `TRUE`, break down results by month. Default: `FALSE`.
`metric_fob`	Logical. Include FOB value (US$). Default: `TRUE`.
`metric_kg`	Logical. Include net weight (kg). Default: `TRUE`.
`metric_statistic`	Logical. Include statistical quantity. Default: `FALSE`.
`metric_freight`	Logical. Include freight value (US$, imports only). Default: `FALSE`.
`metric_insurance`	Logical. Include insurance value (US$, imports only). Default: `FALSE`.
`metric_cif`	Logical. Include CIF value (US$, imports only). Default: `FALSE`.
`language`	Response language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`verbose`	Logical. Show progress messages. Default: `TRUE`.

Value

A data.frame (or tibble if available) with query results.

Examples

```
## Not run:
# Brazilian exports in 2023, by country
comex_query(
  flow = "export",
  start_period = "2023-01",
  end_period = "2023-12",
  details = "country"
)

# Imports 2023 by NCM + country, filtered by specific countries
comex_query(
  flow = "import",
  start_period = "2023-01",
  end_period = "2023-12",
  details = c("ncm", "country"),
  filters = list(country = c(160, 249)),
  month_detail = TRUE,
  metric_cif = TRUE
)

## End(Not run)
```

`comex_query_city` *Query city-level foreign trade data*

Description

Query the city endpoint of the ComexStat API. City-level data is more aggregated than general data, with fewer available details and metrics.

City information is based on the declarant of exports/imports, not the producer or buyer.

Usage

```
comex_query_city(
  flow = "export",
  start_period,
  end_period,
  details = NULL,
  filters = NULL,
  month_detail = TRUE,
  metric_fob = TRUE,
  metric_kg = TRUE,
  metric_statistic = FALSE,
  language = "en",
  verbose = TRUE
)
```

Arguments

`flow`	Trade flow: `"export"` or `"import"`.
`start_period`	Start period in `"YYYY-MM"` format.
`end_period`	End period in `"YYYY-MM"` format.
`details`	Character vector of detail/grouping fields. Options: **Geographic:** `"country"`, `"state"`, `"city"` **Products:** `"hs6"` (or `"sh6"`), `"hs4"` (or `"sh4"`), `"hs2"` (or `"sh2"`), `"section"`
`filters`	Named list of filters. Example: `list(city = "3550308", state = "26")`
`month_detail`	Logical. If `TRUE`, break down by month. Default: `FALSE`.
`metric_fob`	Logical. Include FOB value (US$). Default: `TRUE`.
`metric_kg`	Logical. Include net weight (kg). Default: `TRUE`.
`metric_statistic`	Logical. Include statistical quantity. Default: `FALSE`.
`language`	Response language: `"pt"`, `"en"`, or `"es"`. Default: `"en"`.
`verbose`	Logical. Show progress messages. Default: `TRUE`.

Details

City-level data differs from general data:

- Full NCM is **not** available (use HS6/SH4/SH2)
- Classifications like CGCE, SITC, and ISIC are **not** available
- Only FOB, KG, and Statistical quantity metrics are available
- Freight, Insurance, and CIF metrics are **not** available

Value

A data.frame (or tibble) with query results.

Examples

```
## Not run:
# Exports from Pernambuco in 2023
comex_query_city(
  flow = "export",
  start_period = "2023-01",
  end_period = "2023-12",
  details = c("country", "state"),
  filters = list(state = 26)
)
```

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

---

comex_sitc *Get SITC/CUCI (Standard International Trade Classification) table*

Description

Returns the CUCI (Classificacao Uniforme para o Comercio Internacional) table from the `/tables/product-categories` endpoint. CUCI is the Portuguese name for SITC (Standard International Trade Classification).

Usage

```
comex_sitc(
  language = "en",
  search = NULL,
  add = NULL,
  page = NULL,
  per_page = NULL,
  verbose = FALSE
)
```

Arguments

language	Language: "pt", "en", or "es". Default: "en".
search	Optional search term to filter results.
add	Optional related table to include (e.g. "ncm").
page	Page number for pagination. Default: NULL.
per_page	Number of results per page. Default: NULL.
verbose	Logical. Show progress messages. Default: FALSE.

Value

A data.frame with CUCI/SITC codes and descriptions.

Examples

```
## Not run:
# All CUCI/SITC classifications
comex_sitc()

# Search for products
comex_sitc(search = "carne")

## End(Not run)
```

`comex_states` *Get Brazilian states (UF) table*

Description

Returns the Brazilian states table with codes and names.

Usage

```
comex_states(verbose = FALSE)
```

Arguments

`verbose`	Logical. Default: `FALSE`.

Value

A data.frame with state codes and names.

Examples

```
## Not run:
comex_states()

## End(Not run)
```

`comex_state_detail` *Get state details*

Description

Returns details for a specific Brazilian state.

Usage

```
comex_state_detail(uf_id, verbose = FALSE)
```

Arguments

`uf_id`	State code (e.g. `26` for Pernambuco).
`verbose`	Logical. Default: `FALSE`.

Value

A list with state details.

Examples

```
## Not run:
comex_state_detail(26)

## End(Not run)
```

---

comex_transport_modes *Get transport modes table*

Description

Returns the transport modes table with codes and names.

Usage

```
comex_transport_modes(verbose = FALSE)
```

Arguments

verbose	Logical. Default: FALSE.

Value

A data.frame with transport mode codes and names.

Examples

```
## Not run:
comex_transport_modes()

## End(Not run)
```

---

comex_transport_mode_detail *Get transport mode details*

Description

Returns details for a specific transport mode.

Usage

```
comex_transport_mode_detail(mode_id, verbose = FALSE)
```

Arguments

`mode_id`	Transport mode code (e.g. `5` for maritime).
`verbose`	Logical. Default: `FALSE`.

Value

A list with transport mode details.

Examples

```
## Not run:
comex_transport_mode_detail(5)

## End(Not run)
```

Index