# Package ‘comtradr’

February 6, 2024

**Title**  Interface with the United Nations 'Comtrade' API

**Version**  0.4.0.0

**Maintainer**  Paul Bochtler &lt;paulbochtler.gh@gmail.com&gt;

**Description**  Interface with and extract data from the United Nations 'Comtrade' API &lt;https://comtradeplus.un.org/&gt;. 'Comtrade' provides country level shipping data for a variety of commodities, these functions allow for easy API query and data returned as a tidy data frame.

**Depends**  R (&gt;= 4.1.0)

**Imports**  lifecycle, fs, readr, askpass, cli, httr2, rlang, stringr, poorman, lubridate, purrr, rappdirs, memoise, cachem

**Suggests**  covr, dplyr, ggplot2, httptest2, knitr, rmarkdown, spelling, testthat (&gt;= 3.0.0), callr

**License**  GPL-3

**Encoding**  UTF-8

**LazyData**  true

**RoxygenNote**  7.2.3

**URL**  https://docs.ropensci.org/comtradr/, https://github.com/ropensci/comtradr

**BugReports**  https://github.com/ropensci/comtradr/issues

**NeedsCompilation**  no

**VignetteBuilder**  knitr

**Config/testthat/edition**  3

**Language**  en-US

**Author**  Paul Bochtler [aut, cre, cph] (&lt;https://orcid.org/0000-0002-9146-6185&gt;), Harriet Goers [aut], Chris Muir [aut], Alicia Schep [rev] (&lt;https://orcid.org/0000-0002-3915-0618&gt;), Alicia reviewed the package for rOpenSci, see https://github.com/ropensci/onboarding/issues/141), Rafael Hellwig [rev] (&lt;https://orcid.org/0000-0002-3092-3493&gt;), Rafael
reviewed the package for rOpenSci, see https://github.com/ropensci/onboarding/issues/141), Ernest Guevarra [rev] (<https://orcid.org/0000-0002-4887-4415>, Ernest reviewed the package for rOpenSci, see https://github.com/ropensci/software-review/issues/613), Nicholas Potter [rev] (<https://orcid.org/0000-0002-3410-3732>, Nicholas reviewed the package for rOpenSci, see https://github.com/ropensci/software-review/issues/613), Juergen Amann [ctb]

Repository CRAN

Date/Publication 2024-02-06 18:00:02 UTC

R topics documented:

country_codes .................................................. 2
ct_commodity_db_type ....................................... 3
ct_commodity_lookup ........................................ 4
country_codes lookup ....................................... 5
ct_get_data ..................................................... 6
ct_get_ref_table ............................................... 9
ct_get_remaining_hourly_queries .......................... 10
country_codes get_reset_time ................................ 11
country_codes pretty_cols .................................. 11
country_codes register_token ............................... 12
ct_search ....................................................... 12
country_codes update_databases ................................ 13
country_codes use_pretty_cols .............................. 13
country_codes get_primary_comtrade_key .................. 14
country_codes set_primary_comtrade_key ................. 14

Index 16

country_codes  

Description

A full dataset of all reporter and partner codes available in the UN Comtrade database.

Usage

country_codes
**ct_commodity_db_type**

**Format**

country_codes A dataframe with 312 rows and eight columns:

- **id** Unique country code.
- **country** Name of the country (in English).
- **iso_3** The country’s ISO 3 code.
- **entry_year** The country’s entry into the international system or 1900 (whichever is largest).
- **exit_year** The country’s exit from the international system, if applicable.
- **group** Indicates whether the entity is a group of countries. For example, ASEAN or the European Union.
- **reporter** Indicates whether the country is a reporter in the UN Comtrade database.
- **partner** Indicates whether the country can be reported on by others in the UN Comtrade database. Not all partners are reporters. For example, the World cannot report its trade values.

**Source**


**Description**

This function is deprecated. There is currently no alternative for this function. [Superseded]

**Usage**

ct_commodity_db_type(...)

**Arguments**

... Used to catch all possible arguments that users have supplied to this function.

**Value**

depreciation error

**Examples**

# no examples because only legacy function
ct_commodity_lookup  

UN Comtrade commodities database query

Description

The Comtrade API requires that searches for specific commodities be done using commodity codes. This is a helper function for querying the Comtrade commodity database. It takes as input a vector of commodities or commodity codes. Output is a list or vector of commodity descriptions or codes associated with the input search_terms. For use with the UN Comtrade API, full API docs can be found at https://unstats.un.org/wiki/display/comtrade/

Usage

ct_commodity_lookup(
  search_terms,
  return_code = FALSE,
  commodity_classification = "HS",
  type = "goods",
  return_char = FALSE,
  verbose = TRUE,
  ignore.case = TRUE,
  update = FALSE,
  ...
)

Arguments

search_terms  Commodity names or commodity codes, as a char or numeric vector.
return_code   Logical, if set to FALSE, the function will return a set of commodity descriptions along with commodity codes (as a single string for each match found), if set to TRUE it will return only the commodity codes. Default value is FALSE.
commodity_classification
type         The type of returned trade data. Possible values: ’goods’ for trade in goods, ’services’ for trade in services. Default: ‘goods’.
return_char  Logical, if set to FALSE, the function will return the matches as a named list, if set to TRUE it will return them as a character vector. Default value is FALSE.
verbose      Logical, if set to TRUE, a warning message will print to console if any of the elements of input "search_terms" returned no matches (message will indicate which elements returned no data). Default is TRUE.
ignore.case  logical, to be passed along to arg ignore.case within grepl. Default value is TRUE.
update       If TRUE, downloads possibly updated reference tables from the UN. Default: FALSE.
...          additional args to be passed along to grepl.
Details
This function uses regular expressions (regex) to find matches within the commodity DB. This means it will treat as a match any commodity description that contains the input search term. For more on using regex within R, see https://stat.ethz.ch/R-manual/R-devel/library/base/html/regex.html

Value
A list or character vector of commodity descriptions and/or commodity codes that are matches with the elements of "search_terms".

See Also
grepl

Examples
comtradr::ct_commodity_lookup("wine")

Description
This function is deprecated. You can use country_codes to return a dataset with all possible country codes, but in general the specification of iso 3 codes makes a look-up unnecessary. [Suspended]

Usage
ct_country_lookup(...)

Arguments
... Used to catch all possible arguments that users have supplied to this function.

Value
depreciation error

Examples
# no examples because only legacy function
ct_get_data  

Get trade data from the UN Comtrade API

Description

This function queries the UN Comtrade API to retrieve international trade data. It allows for detailed specification of the query, including the type of data (goods or services), frequency (annual or monthly), commodity classification, flow direction, and more. By providing everything for certain parameters, you can query all possible values. The function is opinionated in that it already verifies certain parameters for you and is more than a pure wrapper around the API.

Usage

ct_get_data(
    type = "goods",
    frequency = "A",
    commodity_classification = "HS",
    commodity_code = "TOTAL",
    flow_direction = c("Import", "Export", "Re-export", "Re-import"),
    reporter = "all_countries",
    partner = "World",
    start_date = NULL,
    end_date = NULL,
    process = TRUE,
    tidy_cols = TRUE,
    verbose = FALSE,
    primary_token = get_primary_comtrade_key(),
    mode_of_transport = "TOTAL modes of transport",
    partner_2 = "World",
    customs_code = "C00",
    update = FALSE,
    requests_per_second = 10/60,
    extra_params = NULL,
    cache = FALSE
)

Arguments

type  The type of returned trade data. Possible values: 'goods' for trade in goods, 'services' for trade in services. Default: 'goods'.

frequency  The frequency of returned trade data. Possible values: 'A' for annual data, 'M' for monthly data. Default: 'A'.

commodity_classification  The trade classification scheme. Possible values for goods: c('HS', 'S1', 'S2', 'S3', 'S4', 'S5', 'B4'), for services: c('EB02', 'EB10', 'EB10S', 'EB'). Default: 'HS'.

commodity_code  The commodity code(s) or everything for all possible codes. See contradr::ct_get_ref_table('HS') for possible values. Default: 'TOTAL' (sum of all commodities).
flow_direction

The direction of trade flows or everything. Possible values can be found in `ct_get_ref_table('flow_direction')`. These are implemented case-insensitive, 'import' and 'Import' are equivalent. Default: c('import','export','re-export','re-import').

reporter

Reporter ISO3 code(s), everything or all_countries. See `comtradr::country_codes` or `comtradr::ct_get_ref_table('reporter')` for possible values. all_countries returns all countries without aggregates. Everything returns all possible parameters. Default: 'all_countries'.

partner

Partner ISO3 code(s), everything or all_countries. See `comtradr::country_codes` or `comtradr::ct_get_ref_table('partner')` for possible values. all_countries returns all countries without aggregates. Everything returns all possible parameters, incl. aggregates like World. Default: 'World' (all partners as an aggregate).

start_date

The start date of the query. Format: yyyy for yearly, yyyy-mm for monthly.

date

The end date of the query. Format: yyyy for yearly, yyyy-mm for monthly. Max: 12 years after start date for annual data, one year for monthly data.

process

If TRUE, returns a data.frame with results. If FALSE, returns the raw http2 request. Default: TRUE.

tidy_cols

If TRUE, returns tidy column names. If FALSE, returns raw column names. Default: TRUE.

verbose

If TRUE, sends status updates to the console. If FALSE, runs functions quietly. Default: FALSE.

primary_token

Your primary UN Comtrade API token. Default: stored token from `comtradr::set_primary_comtrade_key`.

mode_of_transport

Text code of mode of transport or everything for all possible parameters. See `ct_get_ref_table(dataset_id = 'mode_of_transport')` for possible values. Default: 'TOTAL modes of transport' (TOTAL).

partner_2

Partner 2 ISO3 code(s), everything or all_countries. See `comtradr::country_codes` or `comtradr::ct_get_ref_table('partner')` for possible values. all_countries returns all countries without aggregates. Everything returns all possible parameters, incl. aggregates like World. Default: 'World' (all partners as an aggregate).

customs_code

Customs Code ID or everything for all possible parameters. See `ct_get_ref_table(dataset_id = 'customs_code')` for possible values. Default: 'C00' (TOTAL).

update

If TRUE, downloads possibly updated reference tables from the UN. Default: FALSE.

requests_per_second

Rate of requests per second executed, usually specified as a fraction, e.g. 10/60 for 10 requests per minute, see `req_throttle()` for details.

extra_params

Additional parameters to the API, passed as query parameters without checking. Please provide a named list to this parameter. Default: NULL.

cache

A logical value to determine, whether requests should be cached or not. If set to True, `rappdirs::user_cache_dir()` is used to determine the location of the cache. Use the .Renviron file to set the R_USER_CACHE_DIR in order to change this location. Default: False.
Details

The UN Comtrade database provides a repository of official international trade statistics and relevant analytical tables. It contains annual trade statistics starting from 1988 and monthly trade statistics since 2000 for goods data.

Parameters that accept everything will query all possible values. For example, setting commodity_code = 'everything' will retrieve data for all commodity codes. This can be useful for broad queries but may result in large datasets.

Value

A data.frame with trade data or, if process = F, a httr2 response object.

Examples

# Query goods data for China's trade with Argentina and Germany in 2019
cr_get_data()
  type = "goods",
  commodity_classification = "HS",
  commodity_code = "TOTAL",
  reporter = "CHN",
  partner = c("ARG", "DEU"),
  start_date = "2019",
  end_date = "2019",
  flow_direction = "Import",
  partner_2 = "World",
  verbose = TRUE
)

# Query all commodity codes for China's imports from Germany in 2019
ct_get_data()
  commodity_code = "everything",
  reporter = "CHN",
  partner = "DEU",
  start_date = "2019",
  end_date = "2019",
  flow_direction = "Import"
)

# Query all commodity codes for China's imports from Germany
# from January to June of 2019
ct_get_data()
  commodity_code = "everything",
  reporter = "CHN",
  partner = "DEU",
  start_date = "2019",
  end_date = "2019",
  flow_direction = "import"
Description

The first time, the function will read from disk, the second time from the environment. In the case of a necessary update the new data will be saved to the environment for the current session. You can use this table to look at the reference tables and if necessary extract respective classification codes by hand. In general we would recommend the function `ct_commodity_lookup` for this purpose. It uses the present function in the backend.

Usage

```r
ct_get_ref_table(dataset_id, update = FALSE, verbose = FALSE)
```

Arguments

- `dataset_id` The dataset ID, which is either partner, reporter or a valid classification scheme.
- `update` If TRUE, downloads possibly updated reference tables from the UN. Default: FALSE.
- `verbose` If TRUE, sends status updates to the console. If FALSE, runs functions quietly. Default: FALSE.

Details

The function allows you to query most possible input parameters that are listed by the Comtrade API. The following dataset_ids are permitted:

- Datasets that contain codes for the `commodity_code` argument. The name is the same as you would provide under `commodity_classification`.
  - 'HS' This is probably the most common classification for goods.
  - 'B4'
  - 'B5'
  - 'EB02'
  - 'EB10'
  - 'EB10S'
  - 'EB'
  - 'S1'
  - 'S2'
  - 'S3'
  - 'S4'
  - 'SS'
- 'reporter'
- 'partner'
ct_get_remaining_hourly_queries

- 'mode_of_transport'
- 'customs_code'
- 'flow_direction'

Value

a tidy dataset with a reference table

Examples

```r
## get HS commodity table
cr_get_ref_table("HS")

## get reporter table
cr_get_ref_table("reporter")
```

Description

This function is deprecated. There is no more reset time, as the upper limit of 250 calls per day is enforced daily. [Superseded]

Usage

```r
cr_get_remaining_hourly_queries(...)
```

Arguments

... Used to catch all possible arguments that users have supplied to this function.

Value

depreciation error

Examples

```r
# no examples because only legacy function
```
Description
This function is deprecated. There is no more reset time, as the upper limit of 250 calls per day is enforced daily. [Superseded]

Usage
ct_get_reset_time(...)

Arguments
...           Used to catch all possible arguments that users have supplied to this function.

Value
depreciation error

Examples
# no examples because only legacy function

Description
A data.frame with a matched list of tidy and untidy column names for the results.

Usage
country_codes A dataframe with 47 rows and twi columns:

  to  tidy columns
  from original column names
Description
This function is deprecated. Please use set_primary_comtrade_key() instead. [Superseded]

Usage
ct_register_token(...)

Arguments
... Used to catch all possible arguments that users have supplied to this function.

Value
depreciation error

Examples
# no examples because only legacy function

Description
This function is deprecated. Please use ct_get_data() instead. [Superseded]

Usage
ct_search(...)

Arguments
... Used to catch all possible arguments that users have supplied to this function.

Value
depreciation error

Examples
# no examples because only legacy function
ct_update_databases

Description

This function is deprecated. Please use update parameter in the main ct_get_data function instead. [Superseded]

Usage

crct_update_databases(...)ct_update_databases

Arguments

... Used to catch all possible arguments that users have supplied to this function.

Value
depreciation error

Examples

# no examples because only legacy function

crct_use_pretty_cols

Description

This function is deprecated. Please use the process argument in the main function instead. [Superseded]

Usage

crct_use_pretty_cols(...)ct_use_pretty_cols

Arguments

... Used to catch all possible arguments that users have supplied to this function.

Value
depreciation error
Examples

# no examples because only legacy function

---

### get_primary_comtrade_key

**get_primary_comtrade_key**

---

#### Description

If you would like your Comtrade API key to persist in between sessions, use `usethis::edit_r_environ()` to add the env variable COMTRADE_PRIMARY to your environment file.

#### Usage

```r
get_primary_comtrade_key()
```

#### Value

Gets your primary comtrade key from the environment var COMTRADE_PRIMARY

#### Examples

```r
## get API key
get_primary_comtrade_key()
```

---

### set_primary_comtrade_key

**Set your primary Comtrade API key in the environment variable**

---

#### Description

If you would like your Comtrade API key to persist in between sessions, use `usethis::edit_r_environ()` to add the env variable COMTRADE_PRIMARY to your environment file.

#### Usage

```r
set_primary_comtrade_key(key = NULL)
```

#### Arguments

- **key**
  
  Provide your primary comtrade key
set_primary_comtrade_key

Value

Saves your comtrade primary key in the environment.

Examples

```python
## set API key
set_primary_comtrade_key("xxxxxc678ca4dbxxxxxx8285r3")
```
Index

* datasets
  country_codes, 2
  ct_pretty_cols, 11

country_codes, 2
ct_commodity_db_type, 3
country_commodity_lookup, 4
country_lookup, 5
ct_get_data, 6
country_country_lookup, 5
country_get_ref_table, 9
country_get_remaining_hourly_queries, 10
ct_get_reset_time, 11
country_pretty_cols, 11
country_register_token, 12
country_search, 12
country_update_databases, 13
country_use_pretty_cols, 13

get_primary_comtrade_key, 14
grepl, 4, 5

set_primary_comtrade_key, 14