Package: tinycensus (via r-universe)

May 9, 2026

Title Lightweight Interface to the US Census Bureau API

Version 0.0.0.9000

Description Download data from the United States Census Bureau API <https://www.census.gov/data/developers/data-sets.html>. The package provides lightweight, product-specific interfaces for ACS, decennial census data, PEP, CBP, migration flows, and time-series datasets. It also uses Census metadata endpoints to discover datasets, variables, tables, and supported geographies, with optional spatial output joined through 'tinytiger'.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.4.0)

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

URL https://christophertkenny.com/tinycensus/, https://github.com/christopherkenny/tinycensus

BugReports https://github.com/christopherkenny/tinycensus/issues

Imports cli, curl, httr2, rappdirs, rlang, sf, tibble, tinytiger, vctrs

Suggests testthat (>= 3.0.0), vcr

Config/testthat/edition 3

Config/Needs/website christopherkenny/ctktemplate

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

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

Date/Publication 2026-05-01 16:15:37 UTC

RemoteUrl https://github.com/christopherkenny/tinycensus

RemoteRef HEAD

RemoteSha d1d338651a55abe1849a4688484108a1e3ad1a2c

Contents

tc_dataset . . . 2
tc_datasets . . . 3
tc_examples . . . 3
tc_geography . . . 4
tc_get_acs . . . 5
tc_get_cbp . . . 6
tc_get_decennial . . . 7
tc_get_flows . . . 9
tc_get_key . . . 10
tc_get_pdb . . . 10
tc_get_pep . . . 12
tc_get_timeseries . . . 14
tc_groups . . . 15
tc_has_key . . . 16
tc_search_variables . . . 16
tc_set_key . . . 17
tc_table_variables . . . 18
tc_tables . . . 18
tc_values . . . 19
tc_variables . . . 19

Index **20**

`tc_dataset` *Retrieve a single Census dataset record*

Description

Retrieve a single Census dataset record

Usage

```
tc_dataset(dataset, year = NULL, refresh = FALSE)
```

Arguments

`dataset`	A Census dataset identifier like `"acs/acs5"`.
`year`	Optional year. When omitted, the latest available year is used.
`refresh`	Should cached metadata be refreshed?

Value

A tibble with one row.

Examples

```
tc_dataset("acs/acs5", 2024, refresh = TRUE)
```

---

tc_datasets *List Census datasets from the discovery catalog*

Description

List Census datasets from the discovery catalog

Usage

```
tc_datasets(year = NULL, family = NULL, available = TRUE, refresh = FALSE)
```

Arguments

year	Optional year filter.
family	Optional dataset family filter, such as "acs" or "pep".
available	Should only available datasets be returned?
refresh	Should cached metadata be refreshed?

Value

A tibble of Census API datasets.

Examples

```
tc_datasets(year = 2024, family = "acs", refresh = TRUE)
```

---

tc_examples *Retrieve Census example query metadata*

Description

Retrieve Census example query metadata

Usage

```
tc_examples(dataset, year = NULL, refresh = FALSE)
```

Arguments

dataset	A Census dataset identifier like "acs/acs5".
year	Optional dataset year.
refresh	Should cached metadata be refreshed?

Value

A list of example query metadata.

Examples

```
tc_examples("acs/acs5", 2024, refresh = TRUE)
```

---

tc_geography *Retrieve Census geography metadata*

Description

Retrieve Census geography metadata

Usage

```
tc_geography(dataset, year = NULL, refresh = FALSE)
```

Arguments

dataset	A Census dataset identifier like "acs/acs5".
year	Optional dataset year.
refresh	Should cached metadata be refreshed?

Value

A tibble of geography metadata.

Examples

```
tc_geography("acs/acs5", 2024, refresh = TRUE)
```

tc_get_acs *Retrieve American Community Survey data*

Description

Retrieve American Community Survey data

Usage

```
tc_get_acs(
  geography = NULL,
  variables = NULL,
  table = NULL,
  year,
  survey = c("acs5", "acs1", "acs3"),
  product = c("detailed", "profile", "subject", "comparison"),
  within = NULL,
  predicates = NULL,
  summary_var = NULL,
  geometry = FALSE,
  keep_geo_vars = FALSE,
  key = tc_get_key(),
  refresh = FALSE,
  cache = TRUE,
  ucgid = NULL,
  geography_vintage = NULL,
  ...
)
```

Arguments

geography	Census geography name.
variables	Optional character vector of variable names.
table	Optional ACS table identifier. Mutually exclusive with variables.
year	ACS year.
survey	ACS survey, one of "acs1", "acs3", or "acs5".
product	ACS product, one of "detailed", "profile", "subject", or "comparison".
within	Optional named list of parent geographies.
predicates	Optional named list of additional predicates.
summary_var	Optional summary variable to append as summary_estimate / summary_moe.
geometry	Should geometry be joined after retrieval?
keep_geo_vars	Should source geometry attributes be retained?
key	Optional Census API key.

refresh	Should cached metadata be refreshed?
cache	Should discovery metadata be cached locally?
ucgid	Optional ucgid predicate.
geography_vintage	Optional geography vintage used for input normalization.
...	Geography values such as state = "NY" or county = "001".

Value

A tibble or sf object.

Examples

```
tc_get_acs(
  year = 2024,
  variables = "B01001_001E",
  geography = "state",
  state = c("NY", "Delaware")
)
```

---

tc_get_cbp *Retrieve County Business Patterns data*

Description

Retrieve County Business Patterns data

Usage

```
tc_get_cbp(
  geography = NULL,
  variables = NULL,
  table = NULL,
  year,
  dataset = "cbp",
  within = NULL,
  predicates = NULL,
  summary_var = NULL,
  geometry = FALSE,
  keep_geo_vars = FALSE,
  key = tc_get_key(),
  refresh = FALSE,
  cache = TRUE,
  ucgid = NULL,
  geography_vintage = NULL,
  ...
)
```

Arguments

geography	Census geography name.
variables	Optional character vector of variable names.
table	Optional table or group identifier. Mutually exclusive with variables.
year	Dataset year.
dataset	A cbp dataset identifier. Defaults to "cbp".
within	Optional named list of parent geographies.
predicates	Optional named list of additional predicates.
summary_var	Optional summary variable to append as summary_estimate / summary_moe.
geometry	Should geometry be joined after retrieval?
keep_geo_vars	Should source geometry attributes be retained?
key	Optional Census API key.
refresh	Should cached metadata be refreshed?
cache	Should discovery metadata be cached locally?
ucgid	Optional ucgid predicate.
geography_vintage	Optional geography vintage used for input normalization.
...	Geography values such as state = "NY" or county = "001".

Value

A tibble or sf object.

Examples

```
tc_get_cbp(
  year = 2021,
  variables = "ESTAB",
  geography = "state",
  state = c("NY", "DE")
)
```

---

tc_get_decennial *Retrieve Decennial Census data*

Description

Retrieve Decennial Census data

Usage

```
tc_get_decennial(
  geography = NULL,
  variables = NULL,
  table = NULL,
  year,
  dataset = "pl",
  within = NULL,
  predicates = NULL,
  summary_var = NULL,
  geometry = FALSE,
  keep_geo_vars = FALSE,
  key = tc_get_key(),
  refresh = FALSE,
  cache = TRUE,
  ucgid = NULL,
  geography_vintage = NULL,
  ...
)
```

Arguments

geography	Census geography name.
variables	Optional character vector of variable names.
table	Optional table identifier. Mutually exclusive with variables.
year	Decennial Census year.
dataset	Decennial dataset path, such as "pl" or "ddhca".
within	Optional named list of parent geographies.
predicates	Optional named list of additional predicates.
summary_var	Optional summary variable to append as summary_estimate / summary_moe.
geometry	Should geometry be joined after retrieval?
keep_geo_vars	Should source geometry attributes be retained?
key	Optional Census API key.
refresh	Should cached metadata be refreshed?
cache	Should discovery metadata be cached locally?
ucgid	Optional ucgid predicate.
geography_vintage	Optional geography vintage used for input normalization.
...	Geography values such as state = "NY" or county = "001".

Value

A tibble or sf object.

Examples

```
tc_get_decennial(
  year = 2020,
  dataset = "pl",
  variables = "P1_001N",
  geography = "county",
  state = "Delaware"
)
```

`tc_get_flows` *Retrieve ACS migration flows*

Description

Retrieve ACS migration flows

Usage

```
tc_get_flows(
  geography,
  state = NULL,
  county = NULL,
  msa = NULL,
  year = 2018,
  variables = NULL,
  breakdown = NULL,
  breakdown_labels = FALSE,
  geometry = FALSE,
  keep_geo_vars = FALSE,
  key = tc_get_key(),
  refresh = FALSE
)
```

Arguments

`geography`	Flows geography.
`state`	Optional state input.
`county`	Optional county input.
`msa`	Optional metropolitan area codes.
`year`	ACS migration flows year.
`variables`	Optional additional variables.
`breakdown`	Optional breakdown variables.
`breakdown_labels`	Should label columns be added for supported coded breakdown variables?

geometry	Should centroid geometry be joined? Use TRUE or "destination" for destination geometry, or "origin" for origin geometry.
keep_geo_vars	Should source geometry attributes for the selected geometry role be retained?
key	Optional Census API key.
refresh	Included for consistency with other retrieval helpers. Flows data are requested directly and do not currently use cached metadata.

Value

A tibble or sf object.

`tc_get_key` *Get the Census API key*

Description

Get the Census API key

Usage

```
tc_get_key()
```

Value

A character scalar, or "" if no key is set.

Examples

```
invisible(tc_get_key())
```

`tc_get_pdb` *Retrieve Census Planning Database data*

Description

Retrieve Census Planning Database data

Usage

```
tc_get_pdb(
  geography = NULL,
  variables = NULL,
  table = NULL,
  year,
  dataset = NULL,
  within = NULL,
  predicates = NULL,
  summary_var = NULL,
  geometry = FALSE,
  keep_geo_vars = FALSE,
  key = tc_get_key(),
  refresh = FALSE,
  cache = TRUE,
  ucgid = NULL,
  geography_vintage = NULL,
  ...
)
```

Arguments

geography	Census geography name.
variables	Optional character vector of variable names.
table	Optional table or group identifier. Mutually exclusive with variables.
year	Dataset year.
dataset	Optional pdb/... dataset path. When omitted, the dataset is inferred from geography.
within	Optional named list of parent geographies.
predicates	Optional named list of additional predicates.
summary_var	Optional summary variable to append as summary_estimate / summary_moe.
geometry	Should geometry be joined after retrieval?
keep_geo_vars	Should source geometry attributes be retained?
key	Optional Census API key.
refresh	Should cached metadata be refreshed?
cache	Should discovery metadata be cached locally?
ucgid	Optional ucgid predicate.
geography_vintage	Optional geography vintage used for input normalization.
...	Geography values such as state = "NY" or county = "001".

Value

A tibble or sf object.

Examples

```
tc_get_pdb(
  year = 2024,
  variables = "Tot_Population_CEN_2020",
  geography = "tract",
  state = "NY",
  county = "061"
)
```

`tc_get_pep` *Retrieve Population Estimates Program data*

Description

Retrieve Population Estimates Program data

Usage

```
tc_get_pep(
  geography = NULL,
  variables = NULL,
  table = NULL,
  year,
  dataset = NULL,
  product = NULL,
  breakdown = NULL,
  within = NULL,
  predicates = NULL,
  breakdown_labels = FALSE,
  summary_var = NULL,
  geometry = FALSE,
  keep_geo_vars = FALSE,
  key = tc_get_key(),
  refresh = FALSE,
  cache = TRUE,
  ucgid = NULL,
  geography_vintage = NULL,
  ...
)
```

Arguments

`geography`	Census geography name.
`variables`	Optional character vector of variable names.
`table`	Optional table or group identifier. Mutually exclusive with `variables`.

year	Dataset year.
dataset	PEP dataset path, such as "population" or "components". A leading "pep/" is optional.
product	Optional PEP product alias. Supported values are "population", "components", "housing", and "characteristics".
breakdown	Optional characteristics breakdown variables. Supported values are "AGE", "AGEGROUP", "SEX", "HISP", and "RACE".
within	Optional named list of parent geographies.
predicates	Optional named list of additional predicates.
breakdown_labels	Should label variables for supported breakdowns be added automatically?
summary_var	Optional summary variable to append as summary_estimate / summary_moe.
geometry	Should geometry be joined after retrieval?
keep_geo_vars	Should source geometry attributes be retained?
key	Optional Census API key.
refresh	Should cached metadata be refreshed?
cache	Should discovery metadata be cached locally?
ucgid	Optional ucgid predicate.
geography_vintage	Optional geography vintage used for input normalization.
...	Geography values such as state = "NY" or county = "001".

Value

A tibble or sf object.

Examples

```
tc_get_pep(
  year = 2021,
  product = "population",
  geography = "state",
  state = c("NY", "Delaware")
)
```

`tc_get_timeseries` *Retrieve Census time-series data*

Description

Retrieve Census time-series data

Usage

```
tc_get_timeseries(
  geography = NULL,
  dataset,
  variables = NULL,
  table = NULL,
  time = NULL,
  year = NULL,
  within = NULL,
  predicates = NULL,
  key = tc_get_key(),
  refresh = FALSE,
  cache = TRUE,
  ucgid = NULL,
  ...
)
```

Arguments

`geography`	Optional Census geography name.
`dataset`	A time-series dataset identifier.
`variables`	Optional character vector of variable names.
`table`	Optional group or table identifier. Mutually exclusive with `variables`.
`time`	Optional timeseries date value, such as `"2024-01"`.
`year`	Optional dataset year. Most timeseries datasets ignore this and resolve through the discovery catalog.
`within`	Optional named list of parent geographies.
`predicates`	Optional named list of filter predicates other than `time`.
`key`	Optional Census API key.
`refresh`	Should cached metadata be refreshed?
`cache`	Should discovery metadata be cached locally?
`ucgid`	Optional `ucgid` predicate.
`...`	Geography values such as `state = "NY"` or `county = "001"`.

Value

A tibble.

Examples

```
tc_get_timeseries(
  dataset = "intltrade/exports/hs",
  variables = "ALL_VAL_MO",
  time = "2024-01",
  predicates = list(CTY_CODE = "2010")
)
```

---

tc_groups *Retrieve raw Census group metadata*

Description

Retrieve raw Census group metadata

Usage

```
tc_groups(dataset, year = NULL, refresh = FALSE)
```

Arguments

dataset	A Census dataset identifier like "acs/acs5".
year	Optional dataset year.
refresh	Should cached metadata be refreshed?

Details

This is a lower-level metadata helper. For most ACS and decennial workflows, tc_tables() is the more natural entry point because Census "groups" usually correspond to user-facing tables.

Value

A tibble of group metadata from the Census API.

Examples

```
tc_groups("acs/acs5", 2024, refresh = TRUE)
```

tc_has_key *Check for a Census API key*

Description

Check for a Census API key

Usage

```
tc_has_key()
```

Value

A logical scalar.

Examples

```
tc_has_key()
```

tc_search_variables *Search Census variable metadata*

Description

Search Census variable metadata

Usage

```
tc_search_variables(
  dataset,
  year = NULL,
  query,
  fields = c("name", "label", "concept"),
  ignore_case = TRUE,
  refresh = FALSE
)
```

Arguments

dataset	A Census dataset identifier like "acs/acs5".
year	Optional dataset year.
query	Search string.
fields	Metadata fields to search.
ignore_case	Should matching ignore case?
refresh	Should cached metadata be refreshed?

Value

A tibble of matching variables.

`tc_set_key` *Set the Census API key*

Description

Adds a Census API key to the current session and, optionally, to a `.Renviron` file.

Usage

```
tc_set_key(key, overwrite = FALSE, install = FALSE, r_env = NULL)
```

Arguments

`key`	Character scalar API key.
`overwrite`	Should an existing `CENSUS_API_KEY` entry in `.Renviron` be overwritten?
`install`	Should the key be added to a `.Renviron` file?
`r_env`	Path to the `.Renviron` file when `install = TRUE`.

Value

Invisibly returns the key.

Examples

```
## Not run:
tc_set_key("YOUR-API-KEY")
tc_set_key(
  "YOUR-API-KEY",
  install = TRUE,
  overwrite = TRUE,
  r_env = "~/.Renviron"
)

## End(Not run)
```

tc_table_variables *Retrieve variables for a Census table*

Description

Retrieve variables for a Census table

Usage

```
tc_table_variables(dataset, table, year = NULL, refresh = FALSE)
```

Arguments

dataset	A Census dataset identifier like "acs/acs5".
table	Table or group identifier.
year	Optional dataset year.
refresh	Should cached metadata be refreshed?

Value

A tibble of variable metadata for the requested table.

tc_tables *Retrieve Census table metadata*

Description

Retrieve Census table metadata

Usage

```
tc_tables(dataset, year = NULL, refresh = FALSE)
```

Arguments

dataset	A Census dataset identifier like "acs/acs5".
year	Optional dataset year.
refresh	Should cached metadata be refreshed?

Details

This is the preferred helper for exploring ACS and decennial table-level metadata. It wraps the Census API's group metadata in a table-oriented interface.

Value

A tibble of table metadata.

tc_values *Retrieve encoded values metadata for a Census variable*

Description

Retrieve encoded values metadata for a Census variable

Usage

```
tc_values(dataset, year = NULL, variable, refresh = FALSE)
```

Arguments

dataset	A Census dataset identifier like "acs/acs5".
year	Optional dataset year.
variable	Variable name.
refresh	Should cached metadata be refreshed?

Value

A tibble of value codes and labels.

tc_variables *Retrieve Census variable metadata*

Description

Retrieve Census variable metadata

Usage

```
tc_variables(dataset, year = NULL, refresh = FALSE)
```

Arguments

dataset	A Census dataset identifier like "acs/acs5".
year	Optional dataset year.
refresh	Should cached metadata be refreshed?

Value

A tibble of variable metadata.

Examples

```
tc_variables("acs/acs5", 2024, refresh = TRUE)
```

Index

tc_dataset, 2
tc_datasets, 3
tc_examples, 3
tc_geography, 4
tc_get_acs, 5
tc_get_cbp, 6
tc_get_decennial, 7
tc_get_flows, 9
tc_get_key, 10
tc_get_pdb, 10
tc_get_pep, 12
tc_get_timeseries, 14
tc_groups, 15
tc_has_key, 16
tc_search_variables, 16
tc_set_key, 17
tc_table_variables, 18
tc_tables, 18
tc_tables(), *15*
tc_values, 19
tc_variables, 19