

Package: mongolstats (via r-universe)

October 3, 2026

Type Package

Title Tidy Client for the Mongolian 'NSO' 'PXWeb' Statistics API

Version 0.3.0

Description A 'tidyverse'-friendly client for the 'PXWeb' API of the National Statistics Office of Mongolia <<https://data.1212.mn/>>. Discover tables and their dimensions, and fetch statistical data as tibbles with English or Mongolian labels. Periods are reported by their labels and can be selected in any of the office's period formats, tables that have moved within the catalogue are found automatically, and large requests are split to stay within the server's limits. Maps and joins by statistical code are provided by the companion package 'mongolmaps'.

Depends R (>= 4.1.0)

Imports cli, curl, dplyr (>= 1.1.0), httr2 (>= 1.1.0), jsonlite, lifecycle, purrr (>= 1.1.0), rlang, stringi, stringr, tibble

Suggests cachem, carrier, covr, forcats, ggplot2, ggrepel, httptest2, knitr, lintr, lubridate, memoise, mirai, mongolmaps, pkgdown, rappdirs, remotes, rmarkdown, roxygen2, scales, sf, stringdist, styler, testthat (>= 3.1.7), tidyr, withr

Remotes temuulene/mongolmaps

VignetteBuilder knitr

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/testthat/edition 3

URL <https://temuulene.github.io/mongolstats/>,
<https://data.1212.mn/pxweb/>

BugReports <https://github.com/temuulene/mongolstats/issues>

License MIT + file LICENSE

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libicu-dev libssl-dev
Repository https://temuulene.r-universe.dev
Date/Publication 2026-10-03 05:25:42 UTC
RemoteUrl https://github.com/temuulene/mongolstats
RemoteRef HEAD
RemoteSha c5cbabe86e41020c64766ef4d71680d088c28e8d

Contents

as_px_query 3
mn_boundaries 3
mn_boundaries_normalize 4
mn_boundary_keys 5
mn_fuzzy_join_by_name 6
mn_join_by_name 7
nso_cache_clear 8
nso_cache_disable 8
nso_cache_enable 9
nso_cache_status 9
nso_data 10
nso_dim_values 11
nso_dims 12
nso_fetch 12
nso_itms_by_sector 13
nso_latest_periods 14
nso_offline_disable 14
nso_offline_enable 15
nso_options 15
nso_package 16
nso_period_date 17
nso_period_seq 18
nso_query 18
nso_rebuild_px_index 19
nso_search 20
nso_sectors 20
nso_subsectors 21
nso_table_meta 22
nso_table_periods 22
nso_tables 23
nso_variables 24

as_px_query	<i>Convert a query to a PXWeb body</i>
-------------	--

Description

Convert a query to a PXWeb body

Usage

```
as_px_query(x, lang = .px_lang())
```

Arguments

x	An nso_query object.
lang	PX language: "en" or "mn" (defaults to current option).

Value

A list suitable to send as JSON body to PXWeb.

Examples

```
q <- nso_query("DT_NSO_0300_001V2", list(Year = "2023"))
body <- as_px_query(q)
```

mn_boundaries	<i>Mongolia administrative boundaries (sf)</i>
---------------	--

Description**[Deprecated]**

mn_boundaries() was deprecated in mongolstats 0.3.0. Use the mongolmaps package instead: [mongolmaps::mn_admin\(\)](#) and its shortcuts (mn_country(), mn_aimags(), mn_soums(), ...) ship boundaries that carry NSO codes, need no download, and join to NSO tables by code with [mongolmaps::mn_join\(\)](#).

mn_boundaries() downloads boundaries for ADM0/ADM1/ADM2 from the geoBoundaries API and returns an sf object (the sf package must be installed). Results are cached in memory for the session.

Usage

```
mn_boundaries(level = c("ADM0", "ADM1", "ADM2"), refresh = FALSE)
```

Arguments

level	One of "ADM0", "ADM1", "ADM2".
refresh	If TRUE, bypass the session cache and download again.

Value

An sf object with polygons for the requested level. Failed downloads raise an error of class `mongolstats_http_error`.

Examples

```
aimags <- mn_boundaries("ADM1")
# ->
if (rlang::is_installed("mongolmaps")) {
  aimags <- mongolmaps::mn_aimags()
}
```

`mn_boundaries_normalize`

Add normalized name columns to boundaries

Description**[Deprecated]**

`mn_boundaries_normalize()` was deprecated in mongolstats 0.3.0 together with `mn_boundaries()`. To match place names in any spelling, use `mongolmaps::mn_match()`.

Adds a `name_std` column to an sf boundary object by transliterating, lowercasing, and stripping special characters from place names.

Usage

```
mn_boundaries_normalize(g, name_col = "shapeName")
```

Arguments

g	sf object from <code>mn_boundaries()</code> .
name_col	Column with English names (default 'shapeName').

Value

An sf object with an additional `name_std` column.

Examples

```
aimags <- mn_boundaries_normalize(mn_boundaries("ADM1"))
# ->
if (rlang::is_installed("mongolmaps")) {
  mongolmaps::mn_match(c("Khuvsgul", "Hovsgol"), to = "name_en")
}
```

mn_boundary_keys	<i>Boundary keys/crosswalk helper</i>
------------------	---------------------------------------

Description

[Deprecated]

`mn_boundary_keys()` was deprecated in mongolstats 0.3.0. Use [mongolmaps::mn_codes\(\)](#), which lists every administrative unit with its NSO code, ISO 3166-2 code, P-code and names.

Returns a lightweight tibble of key columns from the geoBoundaries data, including normalized names, without the full geometry.

Usage

```
mn_boundary_keys(level = "ADM1")
```

Arguments

level	Boundary level.
-------	-----------------

Value

A tibble with key columns from GeoBoundaries and normalized names.

Examples

```
keys <- mn_boundary_keys("ADM1")
# ->
if (rlang::is_installed("mongolmaps")) {
  keys <- mongolmaps::mn_codes("aimag")
}
```

mn_fuzzy_join_by_name *Fuzzy join data to boundaries by name*

Description

[Deprecated]

mn_fuzzy_join_by_name() was deprecated in mongolstats 0.3.0. Use `mongolmaps::mn_join()`, which knows the common spellings of every aimag and soum and reports each fuzzy match it makes.

Performs a fuzzy string-distance join between a data frame and geoBoundaries polygons.

Usage

```
mn_fuzzy_join_by_name(
  data,
  name_col,
  level = "ADM1",
  boundaries = NULL,
  max_distance = 2,
  method = c("osa", "lv", "jw", "dl")
)
```

Arguments

data	Data frame with a name column.
name_col	Column in data containing names.
level	Boundary level.
boundaries	Optional pre-fetched boundaries.
max_distance	Maximum string distance for a match (default 2). For the edit distances ("osa", "lv", "dl") this counts character edits. Jaro-Winkler ("jw") distances lie between 0 and 1, so use a small value such as 0.2; values of 1 or more would accept every pair and are rejected.
method	Distance method passed to <code>stringdist::stringdist</code> . Ignored (base Levenshtein distance is used) when the <code>stringdist</code> package is not installed.

Value

sf with best fuzzy matches joined. Data rows with no boundary within max_distance are dropped with a warning (class `mongolstats_unmatched_names`) naming them; rows with a missing name are dropped silently.

Examples

```
pop_data <- data.frame(aimag = c("Ulanbatar", "Darhan"), pop = c(1500000, 100000))
sf_joined <- mn_fuzzy_join_by_name(pop_data, "aimag", level = "ADM1")
# ->
if (rlang::is_installed("mongolmaps")) {
  sf_joined <- mongolmaps::mn_join(pop_data, by = "aimag", level = "aimag")
}
```

mn_join_by_name	<i>Join data to boundaries by (normalized) names</i>
-----------------	--

Description**[Deprecated]**

mn_join_by_name() was deprecated in mongolstats 0.3.0. Use [mongolmaps::mn_join\(\)](#), which joins by NSO code or by name in any spelling, drops national and regional totals, and takes Ulaanbaatar's figures from region code "5" when a table leaves code "511" empty.

Performs an exact join between a data frame and geoBoundaries polygons using normalized place names.

Usage

```
mn_join_by_name(data, name_col, level = "ADM1", boundaries = NULL)
```

Arguments

data	Data frame with a name column.
name_col	Column in data that contains names to join on.
level	Boundary level, passed to mn_boundaries() if boundaries not provided.
boundaries	Optional pre-fetched boundaries.

Value

An sf object with joined data. Data rows whose name matches no boundary are dropped with a warning (class mongolstats_unmatched_names) naming them.

Examples

```
pop_data <- data.frame(aimag = c("Ulaanbaatar", "Darkhan-Uul"), pop = c(1500000, 100000))
sf_joined <- mn_join_by_name(pop_data, "aimag", level = "ADM1")
# ->
if (rlang::is_installed("mongolmaps")) {
  sf_joined <- mongolmaps::mn_join(pop_data, by = "aimag", level = "aimag")
}
```

nso_cache_clear	<i>Clear cached entries</i>
-----------------	-----------------------------

Description

Clear cached entries

Usage

nso_cache_clear()

Value

Invisibly, TRUE.

Examples

nso_cache_clear()

nso_cache_disable	<i>Disable caching</i>
-------------------	------------------------

Description

Disable caching

Usage

nso_cache_disable()

Value

Invisibly, TRUE.

Examples

nso_cache_disable()

nso_cache_enable	<i>Enable or configure caching</i>
------------------	------------------------------------

Description

Caches PXWeb catalogue listings and table metadata (codebooks) on disk to speed up repeated calls, including across R sessions. Entries are keyed by the PXWeb base URL, database, and language, so changing `mongolstats.px_base_url` or `mongolstats.px_db` never returns another server's metadata. Data fetched with `nso_data()` is not cached. Optionally set a time-to-live (TTL) for cache entries.

Usage

```
nso_cache_enable(dir = NULL, ttl = NULL)
```

Arguments

<code>dir</code>	Directory for cache; defaults to user cache dir.
<code>ttl</code>	Optional TTL in seconds for cached entries (applies to the disk cache). If NULL, entries persist until cleared.

Value

Cache directory path (invisibly).

Examples

```
# Enable caching in a temporary directory (for demo purposes)
cache_dir <- nso_cache_enable(dir = tempdir())

# Check status
nso_cache_status()

# Disable when done
nso_cache_disable()
```

nso_cache_status	<i>Cache status</i>
------------------	---------------------

Description

Report current cache configuration and basic stats.

Usage

```
nso_cache_status()
```

Value

A list with enabled, dir, and has_cache.

Examples

```
nso_cache_status()
```

nso_data	<i>Fetch statistical data for a table (PXWeb)</i>
----------	---

Description

Fetch statistical data for a table (PXWeb)

Usage

```
nso_data(  
  tbl_id,  
  selections,  
  labels = c("none", "en", "mn", "both"),  
  value_name = getOption("mongolstats.value_name", "value"),  
  include_raw = getOption("mongolstats.attach_raw", FALSE)  
)
```

Arguments

tbl_id	Table identifier (e.g., "DT_NSO_0300_001V2").
selections	Named list mapping variable labels (e.g., Year, Sex) to desired codes or labels.
labels	Label handling: "none" (codes only), "en", "mn", or "both". "code" is accepted as an alias for "none" (matching nso_fetch()).
value_name	Name of the numeric value column in the result (default: "value").
include_raw	If TRUE, attach the raw PX payload as attribute px_raw (a list of payloads when the request was split, see below).

Value

A tibble with one column per dimension and a numeric value column.

Time dimensions

NSO codes periods by position: in every table, code "0" is the latest period, so a period's code changes whenever a new one is published. The time dimension (years, quarters, months or dates) is therefore always returned by its label, such as "2024" or "2024-03", whatever labels is; [nso_period_date\(\)](#) converts these labels to dates. Other dimensions are returned as codes unless labels asks for labels.

Select periods by label (Year = "2024"), with [nso_period_seq\(\)](#) (Month = [nso_period_seq\("202401", "202412", by = "M"\)](#)), or with [nso_latest_periods\(\)](#). Periods match whatever format the table uses, so "202403", "2024-03" and "2024M3" select the same month.

Large requests

data.1212.mn answers at most 1,000,000 cells per request. Larger selections are split into several requests and the results combined; set `options(mongolstats.max_cells = )` to change the limit.

Examples

```
# Population of Mongolia by sex in the two latest years
pop <- nso_data(
  tbl_id = "DT_NSO_0300_001V2",
  selections = list(
    Sex = c("Male", "Female"),
    Age = "Total",
    Year = nso_latest_periods("DT_NSO_0300_001V2", n = 2)
  ),
  labels = "en"
)
pop
```

nso_dim_values	<i>List values for a table dimension</i>
----------------	--

Description

Returns codes and optional labels for a specific dimension.

Usage

```
nso_dim_values(tbl_id, dim, labels = c("code", "en", "mn", "both"))
```

Arguments

tbl_id	Table identifier (e.g., "DT_NSO_0300_001V2").
dim	Dimension name or code (case-insensitive; exact match preferred).
labels	One of "code", "en", "mn", or "both" to control returned label columns. "none" is accepted as an alias for "code".

Value

A tibble with at least code; may include label_en and/or label_mn. In offline mode (see [nso_offline_enable\(\)](#)) an empty tibble is returned; failed requests raise an error of class `mongolstats_http_error`.

Examples

```
values <- nso_dim_values("DT_NSO_0300_001V2", "Year")
head(values)
```

nso_dims	<i>List dimensions for a PXWeb table</i>
----------	--

Description

Returns one row per dimension with basic metadata.

Usage

```
nso_dims(tbl_id)
```

Arguments

tbl_id Table identifier (e.g., "DT_NSO_0300_001V2").

Value

A tibble with columns: dim (display name), code (dimension code), is_time (whether it is the table's time dimension; see [nso_table_periods\(\)](#)), and n_values (number of values for the dimension). In offline mode (see [nso_offline_enable\(\)](#)) an empty tibble is returned; failed requests raise an error of class mongolstats_http_error.

Examples

```
dims <- nso_dims("DT_NSO_0300_001V2")
dims
```

nso_fetch	<i>Fetch a query and return a tibble</i>
-----------	--

Description

Executes an nso_query and returns a tidy tibble with one column per dimension and a numeric value column. Use labels to add _en/_mn columns for each dimension.

Usage

```
nso_fetch(
  x,
  labels = c("code", "en", "mn", "both"),
  value_name = getOption("mongolstats.value_name", "value"),
  include_raw = getOption("mongolstats.attach_raw", FALSE)
)
```

Arguments

x	An nso_query object.
labels	One of "code", "en", "mn", or "both". "none" is accepted as an alias for "code" (matching <code>nso_data()</code>). Defaults to the <code>mongolstats.default_labels</code> option, like <code>nso_data()</code> .
value_name	Name of the numeric value column in the result (default: "value").
include_raw	If TRUE, attach the raw PX payload as attribute <code>px_raw</code> .

Value

A tibble.

Examples

```
q <- nso_query("DT_NS0_0300_001V2", list(Year = "2023"))
data <- nso_fetch(q)
head(data)
```

nso_itms_by_sector	<i>List tables under a sector or sub-sector (PXWeb path)</i>
--------------------	--

Description

Filters the table catalogue to the tables in a given sector or sub-sector, as returned by `nso_sectors()` or `nso_subsectors()`, including the tables in its sub-folders.

Usage

```
nso_itms_by_sector(list_id)
```

Arguments

list_id	Path string from <code>nso_sectors()</code> / <code>nso_subsectors()</code> id.
---------	---

Value

A tibble of the tables in the sector and its sub-folders.

Examples

```
nso_itms_by_sector("Population, household")
```

nso_latest_periods	<i>Most recent periods of a table</i>
--------------------	---------------------------------------

Description

Returns the labels of the latest periods published in a table, ready to use in selections. Combine with [nso_data\(\)](#) to keep a script on the newest data without hard-coding years.

Usage

```
nso_latest_periods(tbl_id, n = 1)
```

Arguments

tbl_id	Table identifier.
n	Number of periods to return.

Value

A character vector of up to n period labels, oldest first; empty in offline mode (see [nso_offline_enable\(\)](#)).

See Also

[nso_table_periods\(\)](#) for every period with its date.

Examples

```
nso_latest_periods("DT_NSO_0300_001V2", n = 3)

# Fetch the latest year without hard-coding it
nso_data(
  "DT_NSO_0300_001V2",
  list(Year = nso_latest_periods("DT_NSO_0300_001V2"), Sex = "Total", Age = "Total")
)
```

nso_offline_disable	<i>Disable offline mode</i>
---------------------	-----------------------------

Description

Disable offline mode

Usage

```
nso_offline_disable()
```

Value

Invisibly, TRUE.

Examples

```
nso_offline_disable()
```

nso_offline_enable	<i>Enable offline mode</i>
--------------------	----------------------------

Description

When enabled, HTTP requests are prevented and functions that require the network will raise a clear offline error. Cached metadata can still be used if already available via `nso_cache_enable()`.

Usage

```
nso_offline_enable()
```

Value

Invisibly, TRUE.

Examples

```
# Enable offline mode
nso_offline_enable()

# Check the option was set
getOption("mongolstats.offline")

# Disable to restore normal operation
nso_offline_disable()
```

nso_options	<i>Set or get mongolstats options</i>
-------------	---------------------------------------

Description

Convenience wrapper around `base::options()` for mongolstats. Values are validated before anything is set: for example `mongolstats.lang` must be "en" or "mn", and `mongolstats.timeout` a positive number. Names that are not mongolstats options are set with a warning, which catches typos.

Usage

```
nso_options(...)
```

Arguments

... Named options to set. If empty, returns a named list of current mongolstats options.

Value

Invisibly, the previous values of the options changed, or a list of current values when called with no arguments.

Examples

```
# Get all current mongolstats options
nso_options()

# Set an option (save old value for restoration)
old <- nso_options(mongolstats.default_labels = "en")

# Restore original value
options(old)
```

nso_package

Fetch multiple tables and bind (PXWeb)

Description

Fetch multiple tables and bind (PXWeb)

Usage

```
nso_package(
  requests,
  labels = c("none", "en", "mn", "both"),
  parallel = getOption("mongolstats.parallel", FALSE),
  value_name = getOption("mongolstats.value_name", "value"),
  strict = FALSE
)
```

Arguments

requests	A list of records, each with tbl_id and selections (named list)
labels	Label handling as in nso_data()
parallel	If TRUE, fetch tables in parallel with <code>purrr::in_parallel()</code> (requires the mirai and carrier packages). Start background workers first with <code>mirai::daemons(4)</code> ; without them the tables are fetched one at a time. Workers load the installed copy of mongolstats. Defaults to the <code>mongolstats.parallel</code> option (FALSE).
value_name	Name of the numeric value column in the result (default: "value").
strict	If TRUE, error when any table fails to fetch. If FALSE (default), failed tables are dropped from the result with a warning naming them.

Value

A tibble combining data from all requested tables, with a `tbl_id` column identifying the source table. Tables that failed to fetch are omitted (with a warning) unless `strict = TRUE`.

Examples

```
reqs <- list(
  list(tbl_id = "DT_NSO_0300_001V2", selections = list(Year = "2023"))
)
combined <- nso_package(reqs)
```

nso_period_date	<i>Convert NSO period labels to dates</i>
-----------------	---

Description

NSO tables label periods in several formats, such as "2024", "2024-03", "2016M3", "2024.03", "2024-I" (first quarter) or "2026-07-06". `nso_period_date()` returns the first day of each period, so results can be sorted, filtered and plotted on a date axis. A trailing *, which NSO uses to mark preliminary figures, is ignored.

Usage

```
nso_period_date(x)
```

Arguments

`x` A character vector of period labels, such as the time column returned by `nso_data()`.

Details

NSO's quarterly tables label quarters "2025-1" to "2025-4". When every label in `x` has that form, `x` is read as quarters; otherwise "2025-4" is April. Pass a table's whole time column so this can be decided, or see the frequency column of `nso_table_periods()`.

Value

A Date vector the same length as `x` holding the first day of each period. Labels that are not periods, such as "Total", give NA.

See Also

`nso_table_periods()` for the periods available in a table.

Examples

```
nso_period_date(c("2024", "2024-03", "2016M3", "2024-II", "Total"))
```

nso_period_seq	Create a sequence of periods
----------------	------------------------------

Description

Builds a run of years (YYYY) or months (YYYYMM) for use in selections. Selections match these to a table’s time labels whatever format the table uses, so "202403" selects "2024-03", "2024M3" or "2024.03".

Usage

```
nso_period_seq(start, end, by = c("Y", "M"))
```

Arguments

- start, end Start and end periods as character (YYYY or YYYYMM).
- by 'Y' for yearly or 'M' for monthly.

Value

Character vector of periods.

Examples

```
# Generate yearly sequence
nso_period_seq("2020", "2024", by = "Y")

# Generate monthly sequence
nso_period_seq("202401", "202406", by = "M")
```

nso_query	Create a PXWeb query object
-----------	-----------------------------

Description

Builds a lightweight query object that records a table id and selections. Use [nso_fetch\(\)](#) to execute it, or [as_px_query\(\)](#) to inspect the underlying PXWeb body.

Usage

```
nso_query(tbl_id, selections = list())
```

Arguments

- tbl_id Table identifier, e.g. "DT_NSO_0300_001V2".
- selections Named list mapping dimension labels (e.g., Year, Sex) to desired codes or labels.

Value

An object of class `nso_query`.

Examples

```
# Create a query object (does not require network)
q <- nso_query("DT_NS0_0300_001V2", list(Year = "2023", Sex = "Total"))
print(q)
```

nso_rebuild_px_index	<i>Rebuild PXWeb index and optionally write to a file</i>
----------------------	---

Description

Crawls the PXWeb API to rebuild the table index. If path is provided, the index is written to that file; otherwise only the in-memory index is refreshed.

Usage

```
nso_rebuild_px_index(path = NULL, write = !is.null(path))
```

Arguments

path	Output path for JSON. If NULL (default), no file is written. For package development, use "inst/extdata/px_index.json".
write	Whether to write JSON to path. Defaults to TRUE if path is provided, FALSE otherwise.

Value

A tibble containing the rebuilt table index. If some catalogue requests fail, the index is built from the rest and a warning of class `mongolstats_incomplete_index` names the failed paths.

Examples

```
# Crawls the entire PXWeb catalogue; takes several minutes
## Not run:
idx <- nso_rebuild_px_index()
head(idx)

## End(Not run)
```

nso_search	<i>Search NSO tables</i>
------------	--------------------------

Description

Searches table titles in the catalogue (see [nso_tables\(\)](#)), case-insensitively, in English and Mongolian by default. query is a regular expression unless fixed = TRUE.

Usage

```
nso_search(
  query,
  sector = NULL,
  fields = c("tbl_eng_nm", "tbl_nm"),
  fixed = FALSE
)
```

Arguments

query	Search string (regex, case-insensitive).
sector	Optional sector or sub-sector path (an id from nso_sectors() or nso_subsectors()); only tables in it or its sub-folders are searched.
fields	Character vector of fields to search within.
fixed	If TRUE, match query literally instead of as a regular expression, so characters such as + or (need no escaping.

Value

Tibble of matching tables.

Examples

```
nso_search("population")
nso_search("infant mortality", sector = "Education, health")
nso_search("c++", fixed = TRUE)
```

nso_sectors	<i>List top-level categories (PXWeb NSO root)</i>
-------------	---

Description

Queries the PXWeb API root to return the top-level statistical sectors (e.g., Population, Economy, Environment). Use the returned id column with [nso_subsectors\(\)](#) to drill into sub-categories.

Usage

```
nso_sectors()
```

Value

A tibble with columns id, type, and text. In offline mode (see [nso_offline_enable\(\)](#)) an empty tibble is returned; failed requests raise an error of class `mongolstats_http_error`.

Examples

```
sectors <- nso_sectors()
head(sectors)
```

nso_subsectors	<i>List children for a given path (PXWeb)</i>
----------------	---

Description

Navigates one level deeper in the PXWeb catalogue hierarchy. Pass in a path id obtained from [nso_sectors\(\)](#) or a previous `nso_subsectors()` call to list its children (sub-sectors or tables).

Usage

```
nso_subsectors(subid)
```

Arguments

subid	Path id from <code>nso_sectors()/nso_subsectors()</code> (e.g., 'Population, household' or 'Population, household/l_Population, household').
-------	--

Value

A tibble with columns: id, type, text. In offline mode (see [nso_offline_enable\(\)](#)) an empty tibble is returned; failed requests raise an error of class `mongolstats_http_error`.

Examples

```
sectors <- nso_sectors()
nso_subsectors(sectors$id[1])
```

nso_table_meta	<i>Table metadata as per-dimension codebooks</i>
----------------	--

Description

Returns a tibble with one row per dimension and a codes list-column, where each element is a tibble of codes and labels (code, label_en, label_mn). Useful for manual query assembly and for inspecting available categories per dimension.

Usage

```
nso_table_meta(tbl_id)
```

Arguments

tbl_id	Table identifier (e.g., "DT_NSO_0300_001V2").
--------	---

Value

A tibble with columns: dim (display name), code (dimension code), is_time (whether it is the table's time dimension), n_values (integer), and codes (list of tibbles). In offline mode (see [nso_offline_enable\(\)](#)) an empty tibble is returned; failed requests raise an error of class `mongolstats_http_error`.

Examples

```
meta <- nso_table_meta("DT_NSO_0300_001V2")
meta
```

nso_table_periods	<i>Periods available in a table</i>
-------------------	-------------------------------------

Description

Finds the table's time dimension (years, quarters, months or dates) and lists its periods from oldest to newest.

Usage

```
nso_table_periods(tbl_id)
```

Arguments

tbl_id	Table identifier.
--------	-------------------

Details

NSO codes periods by position: code "0" is the latest period, so a period's code changes whenever a new one is published. Use the `label` column (or `nso_period_seq()`) in selections, not code.

Value

A tibble with one row per period, oldest first, and columns `code` (the positional PXWeb code), `label` (as shown by NSO, e.g. "2024" or "2024-03"), `date` (first day of the period, see `nso_period_date()`) and `frequency` ("year", "quarter", "month" or "day"). Periods whose label is not a recognisable date keep their catalogue order at the end. The tibble has no rows when the table has no time dimension or in offline mode (see `nso_offline_enable()`). Unknown tables and failed requests raise an error.

See Also

`nso_latest_periods()` for the most recent periods.

Examples

```
periods <- nso_table_periods("DT_NSO_0300_001V2")
tail(periods)
```

nso_tables

List NSO tables

Description

Lists every statistical table in the National Statistics Office PXWeb catalogue. The list comes from an index bundled with the package, so it needs no network access; `nso_rebuild_px_index()` refreshes it from the API. Tables that NSO publishes or moves after the index was built are still found by the other functions, which consult the live catalogue.

Usage

```
nso_tables()
```

Value

A tibble with one row per table and the columns `px_path` (catalogue folder), `px_file`, `tbl_id`, `tbl_eng_nm` and `tbl_nm` (English and Mongolian titles), `strt_prd` and `end_prd` (first and last period), `updated` (when NSO last updated the table, as recorded in the index) and `list_id` (same as `px_path`).

See Also

`nso_search()` to find tables by keyword.

Examples

```
tables <- nso_tables()
tables
```

nso_variables	<i>Variables and values of a table</i>
---------------	--

Description

Returns every dimension value of a table, with its code and its labels in English and Mongolian. Use it to find the codes or labels to pass in selections.

Usage

```
nso_variables(tbl_id)
```

Arguments

tbl_id Table identifier (e.g., "DT_NSO_0300_001V2").

Value

A tibble with one row per dimension value and the columns field (dimension name), itm_id (value code), scr_eng and scr_mn (English and Mongolian labels), px_path and px_file. In offline mode (see [nso_offline_enable\(\)](#)) an empty tibble is returned; failed requests raise an error of class mongolstats_http_error.

See Also

[nso_dims\(\)](#) for one row per dimension, [nso_dim_values\(\)](#) for the values of one dimension.

Examples

```
vars <- nso_variables("DT_NSO_0300_001V2")
vars
```

Index

as_px_query, [3](#)
as_px_query(), [18](#)

base::options(), [15](#)

mn_boundaries, [3](#)
mn_boundaries(), [4](#)
mn_boundaries_normalize, [4](#)
mn_boundary_keys, [5](#)
mn_fuzzy_join_by_name, [6](#)
mn_join_by_name, [7](#)
mongolmaps::mn_admin(), [3](#)
mongolmaps::mn_codes(), [5](#)
mongolmaps::mn_join(), [3](#), [6](#), [7](#)
mongolmaps::mn_match(), [4](#)

nso_cache_clear, [8](#)
nso_cache_disable, [8](#)
nso_cache_enable, [9](#)
nso_cache_status, [9](#)
nso_data, [10](#)
nso_data(), [9](#), [13](#), [14](#), [17](#)
nso_dim_values, [11](#)
nso_dim_values(), [24](#)
nso_dims, [12](#)
nso_dims(), [24](#)
nso_fetch, [12](#)
nso_fetch(), [10](#), [18](#)
nso_itms_by_sector, [13](#)
nso_latest_periods, [14](#)
nso_latest_periods(), [10](#), [23](#)
nso_offline_disable, [14](#)
nso_offline_enable, [15](#)
nso_offline_enable(), [11](#), [12](#), [14](#), [21–24](#)
nso_options, [15](#)
nso_package, [16](#)
nso_period_date, [17](#)
nso_period_date(), [10](#), [23](#)
nso_period_seq, [18](#)
nso_period_seq(), [10](#), [23](#)

nso_query, [18](#)
nso_rebuild_px_index, [19](#)
nso_rebuild_px_index(), [23](#)
nso_search, [20](#)
nso_search(), [23](#)
nso_sectors, [20](#)
nso_sectors(), [13](#), [20](#), [21](#)
nso_subsectors, [21](#)
nso_subsectors(), [13](#), [20](#)
nso_table_meta, [22](#)
nso_table_periods, [22](#)
nso_table_periods(), [12](#), [14](#), [17](#)
nso_tables, [23](#)
nso_tables(), [20](#)
nso_variables, [24](#)

purrr::in_parallel(), [16](#)