Package: mongolstats (via r-universe)

October 31, 2026

Type Package

Title Tidy Access to Mongolian Official Statistics and Metadata

Version 0.4.0

Description A 'tidyverse'-friendly client for the public statistical services of the National Statistics Office of Mongolia <https://data.1212.mn/>. Discover tables and their dimensions, and fetch statistical data as 'tibble' data frames 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. Source provenance and raw observation symbols support reproducible exports. Experimental clients provide access to the public data and metadata portals, including official definitions, collection forms, classifications, and reviewed indicator queries. 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), digest, filelock, 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 (>= 0.2.0), pkgdown, remotes, rmarkdown, roxygen2, scales, sf, stringdist, styler, testthat (>= 3.1.7), tidyr, withr

VignetteBuilder knitr

Encoding UTF-8

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/Needs/website pkgdown@2.1.3

NeedsCompilation no

Author Temuulen Enebish [aut, cre, cph], National Statistics Office of Mongolia [cph] (Bundled statistical extracts; see inst/COPYRIGHTS)

Maintainer Temuulen Enebish <temuulen@gmail.com>

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

Date/Publication 2026-10-31 12:49:38 UTC

RemoteUrl https://github.com/cran/mongolstats

RemoteRef HEAD

RemoteSha 465bf78a5640d851af3c5ac827883d3f70f5ee8a

Contents

as_px_query . . . 3
mn_boundaries . . . 4
mn_boundaries_normalize . . . 4
mn_boundary_keys . . . 5
mn_fuzzy_join_by_name . . . 6
mn_join_by_name . . . 7
nso_api_config . . . 8
nso_cache_clear . . . 9
nso_cache_disable . . . 9
nso_cache_enable . . . 10
nso_cache_status . . . 11
nso_catalogue_status . . . 11
nso_check_data . . . 12
nso_citation . . . 13
nso_classification . . . 14
nso_data . . . 14
nso_dim_values . . . 16
nso_dims . . . 16
nso_fetch . . . 17
nso_forms . . . 18
nso_indicator . . . 18
nso_indicator_query . . . 19
nso_indicators . . . 20
nso_itms_by_sector . . . 20
nso_latest_periods . . . 21
nso_metadata . . . 22
nso_metadata_dictionary . . . 23
nso_metadata_search . . . 23
nso_offline_disable . . . 24
nso_offline_enable . . . 25

nso_options . . . 25
nso_package . . . 26
nso_period_date . . . 27
nso_period_seq . . . 28
nso_portal_data . . . 29
nso_portal_datamarts . . . 30
nso_provenance . . . 31
nso_query . . . 31
nso_rebuild_px_index . . . 32
nso_related . . . 33
nso_search . . . 33
nso_sectors . . . 34
nso_subsectors . . . 35
nso_table_info . . . 36
nso_table_meta . . . 36
nso_table_periods . . . 37
nso_tables . . . 38
nso_value_status . . . 38
nso_variables . . . 39
nso_write_manifest . . . 40

Index **41**

`as_px_query` *Convert a query to a PXWeb body*

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 *Mongolia administrative boundaries (sf)*

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 mn_boundaries().
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 *Boundary keys/crosswalk helper*

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 `stringdist::stringdist`. Ignored (base Levenshtein distance is used) when the stringdist 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` *Join data to boundaries by (normalized) names*

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_api_config` *Inspect PXWeb server configuration*

Description

Retrieves the configuration advertised by the current PXWeb base URL and language. Successful results are cached for this R session. A failed refresh returns the previous configuration with `status = "stale"`; without a cache, conservative defaults are returned. Offline mode never makes a request.

Usage

```
nso_api_config(refresh = FALSE)
```

Arguments

`refresh`	Logical. Bypass the session cache and request the configuration again. Has no network effect in offline mode.

Details

`maxCells` limits cells in a data-query result. `maxValues` limits values returned per variable in table metadata; it is not a selection limit for data queries. `maxCalls` and `timeWindow` describe the server's advertised request allowance. The package separately applies at least one second between NSO request attempts, including retries. Calls on one computer coordinate through `mongolstats.rate_limit_dir`; computers sharing a public IP address require additional coordination by the caller.

Value

A list with numeric `maxCells`, `maxValues`, `maxCalls`, and `timeWindow` (seconds), plus `origin` (`"server"`, `"cache"`, or `"defaults"`), `status` (`"available"`, `"partial"`, `"stale"`, `"offline"`, or `"unavailable"`), `base_url`, and `lang`. Invalid or missing individual limits are replaced by conservative defaults and marked partial.

References

PxWeb configuration manual, API feature, pages 74–75.

Examples

```
old <- nso_options(mongolstats.offline = TRUE)
nso_api_config()
options(old)
```

---

nso_cache_clear *Clear cached entries*

Description

Clear cached entries

Usage

```
nso_cache_clear()
```

Value

Invisibly, TRUE.

Examples

```
nso_cache_clear()
```

---

nso_cache_disable *Disable caching*

Description

Disable caching

Usage

```
nso_cache_disable()
```

Value

Invisibly, TRUE.

Examples

```
nso_cache_disable()
```

nso_cache_enable *Enable or configure caching*

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. The default cache uses `tools::R_user_dir()`, is limited to 10 MiB, and removes entries older than seven days. Existing caches in the former `rappdirs` location are not read or deleted automatically.

Usage

```
nso_cache_enable(dir = NULL, ttl = 7 * 24 * 60 * 60)
```

Arguments

`dir`	Directory for cache; defaults to user cache dir.
`ttl`	Time-to-live in seconds for cached entries. Defaults to seven days. `NULL` disables age-based expiry; the size limit still applies.

Value

Cache directory path (invisibly).

Examples

```
# Enable caching in a temporary directory (for demo purposes)
cache_dir <- tempfile("mongolstats-cache-")
nso_cache_enable(dir = cache_dir)

# Check status
nso_cache_status()

# Disable when done
nso_cache_disable()
unlink(cache_dir, recursive = TRUE)
```

nso_cache_status *Cache status*

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_catalogue_status *Inspect the freshness and coverage of the table catalogue*

Description

Reports index coverage and distinct table counts separately from cross-listed catalogue entries. A bundled index with an unknown build time is identified explicitly. Live keyword searches describe a search window rather than a complete catalogue rebuild.

Usage

```
nso_catalogue_status()
```

Value

A one-row tibble with service identity, `entries`, distinct `tables`, `coverage`, `indexed_at`, and the source of the index.

Examples

```
# Inspect the bundled or session catalogue without a network request.
nso_catalogue_status()
```

`nso_check_data` *Inspect statistical results for analytical concerns*

Description

These optional checks report concerns without altering observations or interpreting undocumented source symbols. Supply observation keys, units, expected periods, and geographic levels explicitly. Different measures and populations require substantive review even when these checks find no issue.

Usage

```
nso_check_data(
  data,
  keys = NULL,
  unit = NULL,
  period = NULL,
  expected_periods = NULL,
  geography_level = NULL
)
```

Arguments

`data`	A data frame, preferably an original fetched result.
`keys`	Character vector of columns identifying an observation.
`unit`	Optional column containing units.
`period`	Optional column containing period labels.
`expected_periods`	Optional expected labels, compared literally.
`geography_level`	Optional column containing explicit geographic levels.

Value

A tibble with `check`, `severity`, `message`, and `n` (affected rows or values). An empty tibble means none of the requested checks found an issue; it does not establish semantic comparability.

Examples

```
# Constructed records contain a duplicate key and an unresolved symbol.
observations <- data.frame(
  period = c("2023", "2023"),
  value = c(12, NA_real_),
  value_raw = c("12", ".."),
  value_status = c("observed", "symbol")
)
nso_check_data(
```

```
  observations,
  keys = "period",
  period = "period",
  expected_periods = c("2023", "2024")
)
```

`nso_citation` *Generate source citations for fetched statistics*

Description

Citations describe the source rather than the software license. NSO public statistics are subject to the source's terms of use and attribution rules; other catalog records may have different access and reuse conditions.

Usage

```
nso_citation(x)
```

Arguments

`x`	A fetched result or provenance manifest.

Value

A character vector of citations, one per source record.

References

https://www.nso.mn/en/terms_of_use

Examples

```
# No citation is invented for data without source records.
observations <- data.frame(value = c(12, 15))
nso_citation(observations)
```

nso_classification *Retrieve a versioned metadata classification codebook*

Description

Retrieve a versioned metadata classification codebook

Usage

```
nso_classification(id, include_raw = FALSE)
```

Arguments

id	Numeric metadata classification identifier.
include_raw	Retain the raw response in the metadata attribute.

Value

A tibble with character codes, definitions, parent codes, source, and version. Attributes contain the typed metadata, quality_flags, and completeness: "empty", "incomplete", or "unverified". Nonempty responses do not establish completeness. No cross-service geographic equivalence or administrative vintage is inferred.

Examples

```
codes <- nso_classification(3446)
codes
attr(codes, "completeness")
attr(codes, "metadata")$version
```

nso_data *Fetch statistical data for a table (PXWeb)*

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). Source provenance and observation status are retained separately in every result; inspect them with nso_provenance() and nso_value_status().

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

Larger selections are split into requests under the configured cell limit. nso_api_config() discovers server limits; cached limits and mongolstats.max_cells are respected. The package coordinates NSO request attempts on the same computer, including workers, at a conservative rate.

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 *List values for a table dimension*

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 *List dimensions for a PXWeb table*

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 *Fetch a query and return a tibble*

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 `nso_data()`). Defaults to the `mongolstats.default_labels` option, like `nso_data()`.
`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`.

Value

A tibble.

Examples

```
q <- nso_query("DT_NSO_0300_001V2", list(Year = "2023"))
data <- nso_fetch(q)
head(data)
```

nso_forms *List collection forms embedded in a metadata database record*

Description

List collection forms embedded in a metadata database record

Usage

```
nso_forms(database_id, include_raw = FALSE)
```

Arguments

database_id	A numeric metadata database identifier.
include_raw	Retain the source database response in the result's metadata attribute and embedded raw form fields in records.

Value

A tibble of form references with a records list column of typed embedded records. Records may be partial; use nso_metadata() with type = "form" to retrieve full details. An empty result means the database response contained no forms.

Examples

```
forms <- nso_forms(database_id = 526)
forms[c("type", "id", "name_mn")]
forms$records
```

nso_indicator *Look up a reviewed concept or a source-qualified portal indicator*

Description

Look up a reviewed concept or a source-qualified portal indicator

Usage

```
nso_indicator(id)
```

Arguments

id	Curated concept, alias, or qualified identifier such as "curated:gdp_current". Portal lookups require the explicit "portal:" prefix and access the live or session-cached catalog.

Value

A one-row tibble from nso_indicators(). Unknown or ambiguous IDs raise an error. A portal match does not supply a reviewed PXWeb template.

Examples

```
nso_indicator("gdp_current")
```

---

nso_indicator_query *Build a PXWeb query from a reviewed statistical concept*

Description

Build a PXWeb query from a reviewed statistical concept

Usage

```
nso_indicator_query(id, periods = NULL, geography = NULL)
```

Arguments

id	Reviewed curated concept or alias; see nso_indicator().
periods	Optional character vector of period labels, not positional PXWeb time codes. NULL selects all source periods.
geography	Optional character vector of source geography codes for concepts with a geography dimension. NULL retains the reviewed national total. Codes use the PXWeb scheme and are validated when the query is fetched.

Value

An nso_query() object ready for nso_fetch(). Construction is offline; fetching validates selections against current table metadata. Metadata and caveats can be inspected with nso_indicator().

Examples

```
nso_indicator_query("gdp_current", periods = c("2023", "2024"))
nso_indicator_query("resident_population", periods = "2024")
```

nso_indicators *Discover reviewed statistical concepts or portal indicators*

Description

The curated registry supplies a small set of reviewed PXWeb query templates. Portal discovery is a separate, experimental catalog: an identifier or title shared with PXWeb does not establish statistical equivalence.

Usage

```
nso_indicators(query = NULL, source = c("curated", "portal"), refresh = FALSE)
```

Arguments

query	Optional literal, case-insensitive text to match across names, identifiers, aliases, and descriptions.
source	"curated" for bundled reviewed concepts, or "portal" for the public government portal catalog.
refresh	Refresh the portal catalog from the service. The curated registry is bundled and reviewed with package releases, so TRUE is not applicable to source = "curated".

Value

A tibble with source-qualified id, source_id, backend, name, unit, query availability, and source metadata. Curated rows also contain definitions, aliases, reviewed selections, caveats and validation dates. Portal rows retain invalid identifiers with queryable = FALSE, report quality flags, and preserve the original registry record in a list column.

Examples

```
nso_indicators("GDP")
nso_indicators("population")
```

nso_itms_by_sector *List tables under a sector or sub-sector (PXWeb path)*

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 nso_sectors()/nso_subsectors() id.

Value

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

Examples

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

---

nso_latest_periods *Most recent periods of a table*

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_metadata` *Retrieve a typed record from the government metadata catalogue*

Description

These experimental interfaces expose the public catalogue at `https://metadata.nso.mn/`. Its numeric identifiers belong to a separate service from PXWeb table identifiers. An `indicator` can describe a questionnaire field rather than a published statistical series.

Usage

```
nso_metadata(type, id, include_raw = FALSE)
```

Arguments

`type`	Entity type: `"database"`, `"table"`, `"indicator"`, `"classification"`, or `"form"`.
`id`	A positive integer identifier in the metadata service.
`include_raw`	Retain the complete parsed response in `$raw`.

Details

Quality flags identify potential inconsistencies and do not correct the upstream record. `include_raw = TRUE` preserves unrecognized fields for inspection. These endpoints have no published stability guarantee. The upstream site's certificate chain may be incomplete on some systems; requests can then fail certificate verification.

Value

An `nso_metadata` list with source-qualified identity, official Mongolian text and available English text, dates, integer years, methodology, access statements, relationships, and quality flags. `$context` retains units, value types, collection sources, dissemination levels, and the responsible department. Its category records preserve source identifiers and official labels. `$date_fields` retains original date and year values, including invalid values, alongside parsed `$dates` and `$years`, even when `include_raw = FALSE`. Missing translations remain `NA`; no translation is generated. Dates use the calendar date supplied by the service, independent of the local time zone. Access statements and licence categories do not establish that the underlying records are publicly downloadable.

Examples

```
field <- nso_metadata("indicator", 12925, include_raw = TRUE)
field$quality_flags
nso_related(field)
```

nso_metadata_dictionary

Retrieve official metadata category dictionaries

Description

Retrieve official metadata category dictionaries

Usage

```
nso_metadata_dictionary(dictionary)
```

Arguments

dictionary	One of "collection-method", "frequency", "security-level", "licence", or "source".

Value

A tibble of source-qualified category identifiers, character codes, and official text. The source_url, retrieved_at, and coverage attributes describe the returned catalogue. A licence or security category describes metadata and does not establish data availability.

Examples

```
nso_metadata_dictionary("frequency")
```

nso_metadata_search

Search or browse the government metadata catalogue

Description

Search or browse the government metadata catalogue

Usage

```
nso_metadata_search(
  query = "",
  type = "database",
  organization = NULL,
  sector = NULL,
  page = 1
)
```

Arguments

query	Search text. A nonempty query uses the site's public search interface and returns its limited result window, with no pagination.
type	Entity type, "all", "organization", or "sector". With an empty query, browse "database", "organization", or "sector".
organization	Numeric organization identifier. Required to browse databases with an empty query.
sector	Numeric sector identifier. Filtering applies only to rows present in the returned response, and rows without the required metadata are excluded. The unresolved_filter_rows attribute counts those rows.
page	Positive page number for database browsing. The package sends this value to the service and preserves the reported pagination without assuming that the server honored the requested page.

Value

A tibble of source-qualified references. Attributes coverage, exhaustive, pagination, retrieved_at, and unresolved_filter_rows describe the scope. exhaustive is always FALSE: a returned page, search window, or response catalogue is not a verified complete inventory.

Examples

```
nso_metadata_search("\u044d\u0440\u04af\u04af\u043b \u043c\u044d\u043d\u0434\u0434", type = "all")
nso_metadata_search(organization = 1, page = 1)
nso_metadata_search(type = "organization")
```

---

nso_offline_disable *Disable offline mode*

Description

Disable offline mode

Usage

```
nso_offline_disable()
```

Value

Invisibly, TRUE.

Examples

```
old <- nso_options(mongolstats.offline = TRUE)
nso_offline_disable()
options(old)
```

nso_offline_enable *Enable offline mode*

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

```
old <- nso_options(mongolstats.offline = FALSE)
# Enable offline mode
nso_offline_enable()

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

# Restore the original setting
options(old)
```

nso_options *Set or get mongolstats options*

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.

Details

mongolstats.request_interval sets the minimum delay between NSO request attempts, with a one-second floor. Processes on the same computer coordinate through mongolstats.rate_limit_dir, which must name the same directory in each process. This does not coordinate other applications or computers sharing a public IP address. mongolstats.portal_base_url and mongolstats.metadata_base_url select the portal and metadata services. mongolstats.max_cells caps data-query cells; mongolstats.max_values describes the metadata value-list limit, rather than a data-query limit.

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 purrr::in_parallel() (requires the mirai and carrier packages). Start background workers first with mirai::daemons(2); without them the tables are fetched one at a time. Workers load the installed copy of mongolstats. Defaults to the mongolstats.parallel 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. nso_provenance() retains a record per successful table plus explicit failure records; nso_value_status() identifies each source observation by tbl_id.

Examples

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

---

nso_period_date *Convert NSO period labels to dates*

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_portal_data` *Fetch a statistical series from the government data portal*

Description

[Experimental]

Retrieves the public aggregate endpoint used by `data.nso.mn`. This service has an unversioned interface and a separate identifier and geography scheme from PXWeb. Similar titles or identifiers do not establish equivalence.

Usage

```
nso_portal_data(
  indicator_id,
  area = "1",
  dimension = "code",
  graph_type = "line",
  periods = NULL,
  labels = c("both", "mn", "en", "code"),
  include_raw = FALSE
)
```

Arguments

`indicator_id`	Portal identifier from `nso_indicators()` with `source = "portal"`, without the `portal:` prefix.
`area`	Portal area parameter, as a nonnegative integer string. The default `"1"` reproduces the verified national GDP request. This is an upstream request parameter, not a PXWeb geography selection.
`dimension`	Portal grouping field: `"code"`, `"code1"`, ..., `"code6"`.
`graph_type`	Portal presentation parameter, such as `"line"`. It is sent to the source endpoint and can affect which records it returns.
`periods`	Optional character vector of exact source period labels. Filtering takes place after retrieval. A trailing `*` is preserved.
`labels`	Official label columns to retain: `"both"`, `"mn"`, `"en"`, or `"code"`. Codes are always retained; missing labels remain missing.
`include_raw`	Attach the original response as attribute `portal_raw`.

Details

The registry supplies source and unit text verbatim when available. It is descriptive source metadata, not an independently validated semantic crosswalk. Narrative summaries are not used to construct observations. Unpadded `YYYY-1` to `YYYY-4` periods are ambiguous between months and quarters, so their dates and frequencies remain missing.

Value

A tibble containing backend, source_id, period, date, frequency, active code dimensions, optional official label columns, value, value_raw, value_status, and registry unit and source. Unknown symbols have status "symbol"; no suppression meaning is inferred. An empty source response returns zero rows; request failures raise errors. A provenance attribute records the request and optional metadata failures.

Examples

```
nso_portal_data("DT_NSO_0000_4_4", periods = "2024")
```

---

nso_portal_datamarts *Discover public aggregate datamarts*

Description

[Experimental]

Lists the portal's public datamart catalog and presentation configuration. A supported developer query contract has not been verified, so this function supplies discovery only. It does not request underlying individual records.

Usage

```
nso_portal_datamarts()
```

Value

A tibble with id, official name, source-reported years, source, is_visible, a configuration list column, and query_supported = FALSE. Year ranges and configuration are source text, not verified data coverage.

Examples

```
nso_portal_datamarts()
```

nso_provenance *Retrieve a data result's provenance manifest*

Description

Provenance records the service, source identifier, resolved URL, selections, retrieval time, source metadata, comments, and a metadata fingerprint. A batch result contains one record per successful table and explicit failure records with request indices. Fetch error conditions retain attempted queries and completed response context; partial tables are never returned as complete. Attributes can be discarded by transformations; save a sidecar with `nso_write_manifest()` before modifying the original result.

Usage

```
nso_provenance(x)
```

Arguments

x	A fetched result, fetch error condition, or a manifest from `nso_read_manifest()`.

Value

A list of class `nso_manifest` with `schema_version`, `records`, and `failures`. An unannotated result has empty records and failures.

Examples

```
# Provenance is not inferred for an ordinary data frame.
observations <- data.frame(value = c(12, 15))
manifest <- nso_provenance(observations)
manifest$records
manifest$failures
```

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_NSO_0300_001V2", list(Year = "2023", Sex = "Total"))
print(q)
```

---

nso_rebuild_px_index *Rebuild PXWeb index and optionally write to a file*

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),
  allow_incomplete = FALSE
)
```

Arguments

path	Output path for JSON. If NULL (default), no file is written. For package development, use "inst/extdata/px_index.json". A companion <path>.metadata.json file preserves coverage and build time, checked against the index's SHA-256 hash and service identity on reload.
write	Whether to write JSON to path. Defaults to TRUE if path is provided, FALSE otherwise.
allow_incomplete	Allow a partial crawl to replace the current index and be written to disk. Defaults to FALSE; incomplete crawls raise a typed error and leave the previous index and output file intact.

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_related` *Inspect relationships in a metadata record*

Description

Inspect relationships in a metadata record

Usage

```
nso_related(record)
```

Arguments

`record`	An `nso_metadata` result.

Value

A tibble of source-qualified references. This function does not make network requests; embedded records may be partial. Retrieve the full record with `nso_metadata()` where its type is supported.

Examples

```
database <- nso_metadata("database", 526)
nso_related(database)
```

`nso_search` *Search NSO tables*

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,
  live = 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.
live	Use the live PXWeb keyword search instead of the bundled title index. The source's search semantics apply; regex matching and fields apply only to the local index. Results carry a catalogue attribute identifying the returned search window. Requires network access.

Value

Tibble of matching tables.

Examples

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

---

nso_sectors *List top-level categories (PXWeb NSO root)*

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` *List children for a given path (PXWeb)*

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 `nso_sectors()`/`nso_subsectors()` (e.g., 'Population, household' or 'Population, household/1_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_info *Inspect table source information*

Description

Fetching a table may supply source information unavailable in its dimension codebook. Inspect a fetched result to see this information. Passing a PXWeb table identifier retrieves its title and dimensions; it does not request observations or infer missing source information.

Usage

```
nso_table_info(x)
```

Arguments

x	A fetched result, a provenance manifest, or one PXWeb table ID.

Value

A tibble with one row per provenance record: service, source ID, title, source statement, update time, unit, URL, retrieval time, and source metadata in a list-column. Fields absent upstream remain missing.

Examples

```
# An unannotated data frame has no source information to report.
observations <- data.frame(value = c(12, 15))
nso_table_info(observations)
nso_table_info(nso_provenance(observations))
```

nso_table_meta *Table metadata as per-dimension codebooks*

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 *Periods available in a table*

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_value_status` *Inspect source values and observation status*

Description

Returns source values separately from numeric observations. Unknown source symbols are preserved without assigning them an undocumented meaning. Status is `"observed"` for finite numbers, `"missing"` for empty or missing values, and `"symbol"` for other source text, including non-finite numbers. Save this table alongside transformed or exported data because ordinary transformations may discard attributes.

Usage

```
nso_value_status(x)
```

Arguments

x	A result from nso_data(), nso_fetch(), nso_package(), or nso_portal_data().

Value

A tibble containing observation keys, value_raw, and value_status, or an empty tibble if source status was not retained.

Examples

```
# Constructed records illustrate numeric values and an unresolved symbol.
observations <- data.frame(
  period = c("2023", "2024"),
  value = c(12, NA_real_),
  value_raw = c("12", ".."),
  value_status = c("observed", "symbol")
)
nso_value_status(observations)

# Ordinary data without retained status have no source-value records.
nso_value_status(data.frame(value = 12))
```

---

nso_variables *Variables and values of a table*

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
```

---

nso_write_manifest *Save or reload a provenance sidecar*

Description

JSON sidecars preserve source identity and query context separately from CSV exports or transformed tibbles. Save nso_value_status() separately for successful results when source symbols matter. Failed fetch manifests retain status from completed chunks when available. Both reading and writing validate the manifest schema. Reading accepts local files; replacement writes preserve the previous file if installing the new file fails.

Usage

```
nso_write_manifest(x, path, overwrite = FALSE)

nso_read_manifest(path)
```

Arguments

x	A fetched result, fetch error condition, or provenance manifest.
path	JSON file path.
overwrite	Whether to replace an existing file. Defaults to FALSE.

Value

nso_write_manifest() returns path invisibly; nso_read_manifest() returns an nso_manifest.

Examples

```
# An unannotated table produces an empty provenance manifest.
observations <- data.frame(value = c(12, 15))
path <- tempfile(fileext = ".json")
nso_write_manifest(observations, path)
saved <- nso_read_manifest(path)
identical(saved, nso_provenance(observations))
unlink(path)
```

Index

as_px_query, 3
as_px_query(), *31*

base::options(), *25*

mn_boundaries, 4
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(), *4*
mongolmaps::mn_codes(), *5*
mongolmaps::mn_join(), *4*, *6*, *7*
mongolmaps::mn_match(), *4*

nso_api_config, 8
nso_api_config(), *15*
nso_cache_clear, 9
nso_cache_disable, 9
nso_cache_enable, 10
nso_cache_status, 11
nso_catalogue_status, 11
nso_check_data, 12
nso_citation, 13
nso_classification, 14
nso_data, 14
nso_data(), *10*, *17*, *21*, *27*, *39*
nso_dim_values, 16
nso_dim_values(), *40*
nso_dims, 16
nso_dims(), *40*
nso_fetch, 17
nso_fetch(), *15*, *19*, *31*, *39*
nso_forms, 18
nso_indicator, 18
nso_indicator(), *19*
nso_indicator_query, 19
nso_indicators, 20
nso_indicators(), *19*, *29*
nso_itms_by_sector, 20
nso_latest_periods, 21
nso_latest_periods(), *15*, *37*
nso_metadata, 22
nso_metadata(), *18*, *33*
nso_metadata_dictionary, 23
nso_metadata_search, 23
nso_offline_disable, 24
nso_offline_enable, 25
nso_offline_enable(), *16*, *17*, *21*, *35*, *37*, *39*
nso_options, 25
nso_package, 26
nso_package(), *39*
nso_period_date, 27
nso_period_date(), *15*, *37*
nso_period_seq, 28
nso_period_seq(), *15*, *37*
nso_portal_data, 29
nso_portal_data(), *39*
nso_portal_datamarts, 30
nso_provenance, 31
nso_provenance(), *15*, *27*
nso_query, 31
nso_query(), *19*
nso_read_manifest *(*nso_write_manifest*)*, 40
nso_read_manifest(), *31*
nso_rebuild_px_index, 32
nso_rebuild_px_index(), *38*
nso_related, 33
nso_search, 33
nso_search(), *38*
nso_sectors, 34
nso_sectors(), *20*, *34*, *35*
nso_subsectors, 35
nso_subsectors(), *20*, *34*
nso_table_info, 36
nso_table_meta, 36
nso_table_periods, 37

nso_table_periods(), *17*, *21*, *27*, *28*
nso_tables, 38
nso_tables(), *33*
nso_value_status, 38
nso_value_status(), *15*, *27*, *40*
nso_variables, 39
nso_write_manifest, 40
nso_write_manifest(), *31*

purrr::in_parallel(), *27*

tools::R_user_dir(), *10*