# Package: tuiksdmx (via r-universe)

November 14, 2026

**Title** Access the Turkish Statistical Institute SDMX REST API

**Version** 0.1.1

**Description** Provides an authenticated, metadata-driven R client for the Turkish Statistical Institute (TURKSTAT) SDMX REST API. Supports dataflow discovery, data structure definitions, concepts, codelists, metadata-based filter validation, SDMX query construction, statistical data retrieval, and observation-based availability exploration. Authentication supports an environment variable or an explicitly supplied API key argument.

**URL** https://github.com/MFatihTuzen/tuiksdmx, https://MFatihTuzen.github.io/tuiksdmx/

**BugReports** https://github.com/MFatihTuzen/tuiksdmx/issues

**License** MIT + file LICENSE

**Depends** R (>= 4.1.0)

**Encoding** UTF-8

**Imports** httr2, xml2, dplyr, purrr, tibble, readr, rlang, digest

**Suggests** testthat (>= 3.1.7), withr, knitr, rmarkdown

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**Config/roxygen2/version** 8.1.1

**NeedsCompilation** no

**Author** Muhammed Fatih Tuzen [aut, cre]

**Maintainer** Muhammed Fatih Tuzen <mfatihtuzen@gmail.com>

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

**Date/Publication** 2026-11-13 18:10:02 UTC

**RemoteUrl** https://github.com/cran/tuiksdmx

**RemoteRef** HEAD

**RemoteSha** b281a7d06c98ce9d8e8330a8e8a2700666fb74c2

# Contents

tuik_attributes . . . 2
tuik_available_codes . . . 3
tuik_build_key . . . 4
tuik_concepts . . . 5
tuik_dataflows . . . 6
tuik_decode_data . . . 7
tuik_dimension_codes . . . 9
tuik_dimensions . . . 10
tuik_explore . . . 10
tuik_filter_dimensions . . . 11
tuik_filter_template . . . 12
tuik_flow_profile . . . 13
tuik_get_codelist . . . 14
tuik_get_concept_scheme . . . 14
tuik_get_data . . . 15
tuik_get_dataflow . . . 17
tuik_get_dsd . . . 18
tuik_key_dimensions . . . 19
tuik_measures . . . 19
tuik_missing_filters . . . 20
tuik_search_concepts . . . 20
tuik_search_dataflows . . . 21
tuik_validate_filters . . . 22

**Index** **24**

---

`tuik_attributes` *Extract attributes from a TÜİK SDMX DSD*

---

## Description

Parses attribute components and their coded representations from a DSD.

## Usage

```
tuik_attributes(dsd)
```

## Arguments

| | |
|---|---|
| `dsd` | XML document returned by `tuik_get_dsd()`. |

## Value

A tibble describing DSD attributes.

`tuik_available_codes` *Retrieve codes observed in a Dataflow for a time window*

## Description

Returns values of a dimension that are actually present in data returned by the selected Dataflow over a bounded period. This is intentionally distinct from `tuik_dimension_codes()`, which returns the theoretical Codelist domain.

## Usage

```
tuik_available_codes(
  profile,
  dimension,
  start_period,
  end_period,
  filters = NULL,
  labels = TRUE,
  api_key = NULL,
  query_mode = c("auto", "all"),
  allow_all = FALSE
)
```

## Arguments

| | |
|---|---|
| `profile` | A `tuik_flow_profile` object. |
| `dimension` | Dimension ID whose observed values should be returned. |
| `start_period` | Required start period. |
| `end_period` | Required end period. |
| `filters` | Optional named list of dimension filters. Filter values are validated against Codelists, and reapplied locally to the returned data. |
| `labels` | Logical. If TRUE, attach Turkish and English Codelist labels when the dimension references a Codelist. |
| `api_key` | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |
| `query_mode` | `"auto"` (default) uses a complete server-side SDMX key when possible. `"all"` always requests all series for the period. The `all` path requires `allow_all = TRUE` in either mode. |
| `allow_all` | Logical. Explicitly authorizes a potentially large `all` download if filters are partial/absent or `query_mode = "all"`. This does NOT impose a byte, row, memory, or timeout limit. |

## Details

By default `query_mode = "auto"` uses a server-side key for complete filters. Partial or absent filters require the explicit `allow_all = TRUE` opt-in because the service may return a large `all` response. `query_mode = "all"` also requires `allow_all = TRUE`. Both period bounds are required. Period bounds do not impose a limit on response size or memory usage. This is observed data availability, not an SDMX ContentConstraint.

## Value

A tibble with observed code values, row counts and optional labels.

## Examples

```
## Not run:
tuik_available_codes(
  profile,
  dimension = "COICOP_2018",
  start_period = "2026-01",
  end_period = "2026-03",
  allow_all = TRUE  # No filters: explicit broad-download opt-in
)

tuik_available_codes(
  profile,
  dimension = "COICOP_2018",
  start_period = "2026-01",
  end_period = "2026-03",
  filters = list(FREQ = "M", REF_AREA = "TR"),
  allow_all = TRUE  # Partial filters: downloaded data are filtered locally
)

## End(Not run)
```

---

# tuik_build_key *Build an SDMX data key*

---

## Description

Builds a series key using the exact dimension order defined by the DSD.

## Usage

```
tuik_build_key(profile, filters)
```

## Arguments

| | |
|---|---|
| `profile` | A `tuik_flow_profile` object. |
| `filters` | Named list of dimension selections. |

## Details

Each dimension may contain:

- one value, e.g. "M"
- multiple values, e.g. c("01", "02")

Multiple values are joined using the SDMX "+" separator.

At the current stage, all series dimensions must be supplied explicitly. This is intentional because blank-segment wildcard behaviour has not been reliably confirmed for the TÜİK service.

## Value

Character string containing the SDMX key.

## Examples

```
## Not run:
# Requires TUIK_SDMX_API_KEY in the environment.
library(tuiksdmx)
profile <- tuik_flow_profile("DF_TUFE_SDMX_TT01")

key <- tuik_build_key(
  profile,
  filters = list(
    REF_AREA = "TR",
    FREQ = "M",
    SINIFLAMA_DUZEYI = "2",
    DEGISIM = "1",
    OZEL_KAPSAM_TUFE = "_Z",
    BASE_PER = "2025",
    YAYIM_DONEMI = "2026_01",
    COICOP_1999 = "_Z",
    COICOP_2018 = c("01", "02"),
    INDICATOR = "F_TFE"
  )
)

## End(Not run)
```

---

# tuik_concepts *Retrieve concepts used by a TÜİK Dataflow*

---

## Description

Resolves the concepts referenced by dimensions, measures and attributes in a flow profile and adds their Turkish/English labels and descriptions.

### Usage

```
tuik_concepts(profile, refresh = FALSE, api_key = NULL)
```

### Arguments

| | |
|---|---|
| profile | A tuik_flow_profile object. |
| refresh | Logical. If TRUE, refresh ConceptScheme metadata. |
| api_key | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |

### Value

A tibble with component-to-concept mappings and semantic metadata.

---

## tuik_dataflows *List TÜİK SDMX Dataflows*

---

### Description

Retrieves the current Dataflow catalogue published by TÜİK.

### Usage

```
tuik_dataflows(
  agency = "TR",
  version = "latest",
  refresh = FALSE,
  api_key = NULL
)
```

### Arguments

| | |
|---|---|
| agency | Dataflow agency. Defaults to "TR". |
| version | Version selector. Defaults to "latest". |
| refresh | Logical. If TRUE, ignore the cached catalogue and retrieve a fresh copy from the service. |
| api_key | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |

## Details

The returned table contains:

- Dataflow agency, ID and version
- Final/non-final status
- Turkish and English names
- Referenced DSD agency, ID and version

Results are cached in memory for the current R session. Use `refresh = TRUE` to force a new request to the TÜİK SDMX service.

## Value

A tibble with one row per Dataflow.

## Examples

```
## Not run:
flows <- tuik_dataflows()

# Force a fresh download
flows <- tuik_dataflows(refresh = TRUE)

## End(Not run)
```

---

`tuik_decode_data` *Add SDMX code labels to an observation table*

---

## Description

Joins dimension code labels from the Codelists referenced by a Dataflow profile. Original SDMX dimension codes and observations are preserved. The labels describe Codelist definitions, not observed availability.

## Usage

```
tuik_decode_data(
  data,
  profile,
  dimensions = NULL,
  language = c("tr", "en", "both"),
  api_key = NULL
)
```

## Arguments

| | |
|---|---|
| data | A data frame returned by tuik_get_data(). |
| profile | A tuik_flow_profile object. |
| dimensions | NULL (all coded dimensions present in data) or a character vector of dimension IDs to decode. |
| language | One of "tr", "en", or "both". |
| api_key | Optional API key. If NULL, TUIK_SDMX_API_KEY is used. |

## Value

A tibble containing the input columns plus DIMENSION_label_tr and/or DIMENSION_label_en columns. Unknown code labels remain NA.

## Examples

```
## Not run:
# Requires TUIK_SDMX_API_KEY in the environment.
library(tuiksdmx)
profile <- tuik_flow_profile("DF_TUFE_SDMX_TT01")

# Use the complete CPI filter set documented in tuik_get_data().
# Check current metadata before reuse; codes and availability may change.
data <- tuik_get_data(
  profile,
  filters = list(
    REF_AREA = "TR",
    FREQ = "M",
    SINIFLAMA_DUZEYI = "2",
    DEGISIM = "1",
    OZEL_KAPSAM_TUFE = "_Z",
    BASE_PER = "2025",
    YAYIM_DONEMI = "2026_01",
    COICOP_1999 = "_Z",
    COICOP_2018 = "01",
    INDICATOR = "F_TFE"
  ),
  start_period = "2026-01",
  end_period = "2026-01"
)

decoded <- tuik_decode_data(data, profile, dimensions = c("DEGISIM", "FREQ"))

## End(Not run)
```

`tuik_dimension_codes` *Retrieve codes for a DSD dimension*

## Description

Finds the Codelist referenced by a dimension and returns its codes.

## Usage

```
tuik_dimension_codes(
  dimensions,
  dimension,
  search = NULL,
  language = c("both", "tr", "en"),
  limit = 50,
  api_key = NULL
)
```

## Arguments

| | |
|---|---|
| `dimensions` | Dimension table returned by `tuik_dimensions()` or a `tuik_flow_profile` object. |
| `dimension` | Dimension ID, for example "FREQ" or "COICOP_2018". |
| `search` | Optional search term. |
| `language` | Search language: "both", "tr", or "en". |
| `limit` | Maximum number of rows to return. Use NULL for all rows. |
| `api_key` | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |

## Details

Optionally searches within code values and Turkish/English labels.

## Value

A tibble containing the dimension codes.

## Examples

```
## Not run:
tuik_dimension_codes(
  dimensions,
  "FREQ"
)

tuik_dimension_codes(
  dimensions,
```

```
  "COICOP_2018",
  search = "food"
)

## End(Not run)
```

---

`tuik_dimensions` *Extract dimensions from a TÜİK SDMX DSD*

---

## Description

Parses Dimension and TimeDimension components from a DSD.

## Usage

```
tuik_dimensions(dsd)
```

## Arguments

| | |
|---|---|
| `dsd` | XML document returned by `tuik_get_dsd()`. |

## Value

A tibble describing DSD dimensions.

---

`tuik_explore` *Explore a TUIK SDMX Dataflow before downloading observations*

---

## Description

Displays structural components from a Dataflow's DSD without downloading observations. Series dimensions are selected using `filters`; TIME_PERIOD uses `start_period` and `end_period` in `tuik_get_data()`. A specific coded dimension can be inspected on demand. Codelist entries are theoretical, not necessarily observed in a selected time window.

## Usage

```
tuik_explore(
  dataflow,
  dimension = NULL,
  search = NULL,
  language = c("both", "tr", "en"),
  limit = 50,
  api_key = NULL,
  filters = NULL
)
```

### Arguments

| | |
|---|---|
| dataflow | A Dataflow ID or a tuik_flow_profile object. |
| dimension | Optional series dimension ID whose codes to display. |
| search | Optional search text for dimension codes. |
| language | One of "both", "tr", or "en". |
| limit | Maximum codes to display for a selected dimension; default 50. |
| api_key | Optional API key; defaults to TUIK_SDMX_API_KEY. |
| filters | Optional named list of selected series-dimension codes. When supplied without dimension, adds selected_codes and filter_status columns to the DSD overview. This reports completeness only; it does not validate Codelist membership or download observations. Pass the same filters to tuik_get_data(), which validates them before retrieval. |

### Value

Without dimension, a tibble describing DSD components and how to select them. If filters is supplied, also shows selections and missing series dimensions. With dimension, a tibble of Codelist codes and labels.

### Examples

```
## Not run:
tuik_explore("DF_TUFE_SDMX_TT01")
tuik_explore("DF_TUFE_SDMX_TT01", dimension = "DEGISIM")
selections <- list(REF_AREA = "TR", FREQ = "M", DEGISIM = "1")
tuik_explore("DF_TUFE_SDMX_TT01", filters = selections)
data <- tuik_get_data("DF_TUFE_SDMX_TT01", filters = selections,
                      filter_mode = "local", start_period = "2026-01",
                      end_period = "2026-01")

## End(Not run)
```

---

## tuik_filter_dimensions

*Show required filters for a TÜİK SDMX Dataflow*

---

### Description

Returns the dimensions required to build a series key together with their referenced Codelists.

### Usage

```
tuik_filter_dimensions(profile)
```

### Arguments

| | |
|---|---|
| profile | A tuik_flow_profile object. |

## Value

A tibble describing required filter dimensions.

---

`tuik_filter_template` *Create a filter template for a TÜİK SDMX Dataflow*

---

## Description

Creates a named list containing all dimensions required for an SDMX series key. Each value is initially NULL and can be filled by the user.

## Usage

```
tuik_filter_template(profile)
```

## Arguments

| | |
|---|---|
| `profile` | A `tuik_flow_profile` object. |

## Details

TIME_PERIOD is excluded because time filtering is handled separately through startPeriod and endPeriod.

## Value

A named list with one element per series-key dimension.

## Examples

```
## Not run:
# Requires TUIK_SDMX_API_KEY in the environment.
library(tuiksdmx)
profile <- tuik_flow_profile("DF_TUFE_SDMX_TT01")

filters <- tuik_filter_template(profile)

filters$REF_AREA <- "TR"
filters$FREQ <- "M"

## End(Not run)
```

`tuik_flow_profile` *Build a profile for a TÜİK SDMX Dataflow*

## Description

Retrieves a Dataflow, follows its referenced DSD and parses its structural components.

## Usage

```
tuik_flow_profile(
  dataflow_id,
  agency = "TR",
  version = "latest",
  api_key = NULL
)
```

## Arguments

| | |
|---|---|
| `dataflow_id` | Dataflow identifier. |
| `agency` | Dataflow agency. Defaults to "TR". |
| `version` | Dataflow version. Defaults to "latest". |
| `api_key` | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |

## Value

A list with the following components:

- dataflow
- dsd
- dimensions
- measures
- attributes

## Examples

```
## Not run:
profile <- tuik_flow_profile(
  "DF_TUFE_SDMX_TT01",
  version = "1.0"
)

profile$dimensions

## End(Not run)
```

## tuik_get_codelist *Retrieve a TÜİK SDMX Codelist*

### Description

Downloads and parses an SDMX Codelist.

### Usage

```
tuik_get_codelist(
  codelist_id,
  agency,
  version = "latest",
  refresh = FALSE,
  api_key = NULL
)
```

### Arguments

| | |
|---|---|
| codelist_id | Codelist identifier. |
| agency | Codelist agency. |
| version | Codelist version. |
| refresh | Logical. If TRUE, ignore the cache and download the Codelist again. |
| api_key | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |

### Details

Results are cached in memory for the current R session.

### Value

A tibble with code, Turkish label and English label.

## tuik_get_concept_scheme *Retrieve a TÜİK SDMX ConceptScheme*

### Description

Downloads and parses a ConceptScheme. Results are cached in memory for the current R session.

### Usage

```
tuik_get_concept_scheme(
  concept_scheme_id,
  agency,
  version = "latest",
  refresh = FALSE,
  api_key = NULL
)
```

### Arguments

| | |
|---|---|
| concept_scheme_id | ConceptScheme identifier. |
| agency | ConceptScheme agency. |
| version | ConceptScheme version. Defaults to "latest". |
| refresh | Logical. If TRUE, ignore the cache and download again. |
| api_key | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |

### Value

A tibble with Concept IDs and multilingual labels/descriptions.

---

## tuik_get_data *Retrieve data from a TÜİK SDMX Dataflow*

---

### Description

Retrieves all series with `filters = NULL`, or builds a complete SDMX key from supplied filters. Partial filters require explicit permission via `allow_all = TRUE` or `filter_mode = "local"`: the service's `all` data are retrieved first and filters are applied in R. At least one period bound is required unless `allow_unbounded = TRUE`. This is NOT server-side filtering and can download large responses even for a short period. Complete filters are always queried by SDMX key on the server.

### Usage

```
tuik_get_data(
  profile,
  filters = NULL,
  start_period = NULL,
  end_period = NULL,
  parse_observation = TRUE,
  validate = TRUE,
  api_key = NULL,
  allow_unbounded = FALSE,
  labels = c("none", "tr", "en", "both"),
```

```
  label_dimensions = NULL,
  filter_mode = c("reject", "local"),
  allow_all = FALSE
)
```

## Arguments

| | |
|---|---|
| profile | A tuik_flow_profile object or a single Dataflow ID. When an ID is supplied, a profile is resolved automatically with tuik_flow_profile() using the same api_key. For explicit Dataflow agency/version selection, construct a profile first. |
| filters | NULL to request all series, or a named list of key-dimension selections. Partial selections require allow_all = TRUE (or legacy filter_mode = "local") before the service's all data are downloaded. |
| start_period | Optional start period, for example "2026-01". |
| end_period | Optional end period, for example "2026-03". |
| parse_observation | Logical. If TRUE, attempts to convert OBS_VALUE to numeric while preserving all SDMX dimensions and attributes as character columns. |
| validate | Logical. If TRUE, validate supplied filter values against the Codelists referenced by the Dataflow DSD before requesting data. Defaults to TRUE. |
| api_key | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |
| allow_unbounded | Logical. Must be TRUE to permit unfiltered or locally filtered requests with neither period bound supplied. This can download a very large dataset. Does not limit the server response size. |
| labels | Language of optional Codelist labels: "none" (the default), "tr", "en", or "both". Raw SDMX codes remain unchanged. |
| label_dimensions | Optional character vector of dimension IDs to decode when labels is not "none". If NULL, decode all coded dimensions present in the downloaded data. |
| filter_mode | Legacy explicit mode: "reject" (default) or "local". Existing calls remain supported. For new code, prefer allow_all = TRUE to authorize a locally filtered all request. |
| allow_all | Logical, default FALSE. With incomplete filters, TRUE authorizes an all request and applies the specified filters locally. This is the same opt-in used by tuik_available_codes(). It DOES NOT limit transferred bytes, memory, or response time. It does not bypass allow_unbounded when neither period bound is provided. |

## Value

A tibble containing SDMX observations. If labels are requested, additional <DIMENSION>_label_tr and/or <DIMENSION>_label_en columns are appended. Labels identify Codelist-defined values, not observed availability.

## Examples

```
## Not run:
# Requires TUIK_SDMX_API_KEY in the environment.
library(tuiksdmx)
profile <- tuik_flow_profile("DF_TUFE_SDMX_TT01")

data <- tuik_get_data(
  profile,
  filters = list(
    REF_AREA = "TR",
    FREQ = "M",
    SINIFLAMA_DUZEYI = "2",
    DEGISIM = "1",
    OZEL_KAPSAM_TUFE = "_Z",
    BASE_PER = "2025",
    YAYIM_DONEMI = "2026_01",
    COICOP_1999 = "_Z",
    COICOP_2018 = "01",
    INDICATOR = "F_TFE"
  ),
  start_period = "2026-01",
  end_period = "2026-03"
)

## End(Not run)
```

---

# tuik_get_dataflow *Retrieve a single TÜİK SDMX Dataflow*

---

## Description

Retrieves metadata for one Dataflow and its referenced DSD.

## Usage

```
tuik_get_dataflow(
  dataflow_id,
  agency = "TR",
  version = "latest",
  api_key = NULL
)
```

## Arguments

| | |
|---|---|
| dataflow_id | Dataflow identifier. |
| agency | Dataflow agency. Defaults to "TR". |
| version | Dataflow version. Defaults to "latest". |

| | |
|---|---|
| api_key | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |

## Value

A one-row tibble.

## Examples

```
## Not run:
tuik_get_dataflow(
  "DF_TUFE_SDMX_TT01",
  version = "1.0"
)

## End(Not run)
```

---

# tuik_get_dsd *Retrieve a TÜİK SDMX Data Structure Definition*

---

## Description

Retrieves a DSD from the TÜİK SDMX Web Service.

## Usage

```
tuik_get_dsd(dsd_id, agency = "TR", version = "latest", api_key = NULL)
```

## Arguments

| | |
|---|---|
| dsd_id | Data Structure Definition identifier. |
| agency | DSD agency. Defaults to "TR". |
| version | DSD version. Defaults to "latest". |
| api_key | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |

## Value

An xml2 XML document.

## Examples

```
## Not run:
dsd <- tuik_get_dsd(
  "DSD_TUFE",
  version = "1.12"
)

## End(Not run)
```

---

`tuik_key_dimensions` *Return the dimensions used in an SDMX series key*

---

## Description

Extracts ordinary series dimensions from a TÜİK flow profile.

## Usage

```
tuik_key_dimensions(profile)
```

## Arguments

| | |
|---|---|
| `profile` | A `tuik_flow_profile` object. |

## Details

TimeDimension components such as TIME_PERIOD are excluded because they are supplied separately through time query parameters.

## Value

A tibble containing the key dimensions in DSD order.

---

`tuik_measures` *Extract measures from a TÜİK SDMX DSD*

---

## Description

Parses measure components from a DSD.

## Usage

```
tuik_measures(dsd)
```

### Arguments

| | |
|---|---|
| dsd | XML document returned by tuik_get_dsd(). |

### Value

A tibble describing DSD measures.

---

## tuik_missing_filters *Identify missing SDMX key filters*

---

### Description

Compares a filter list with the dimensions required by the DSD.

### Usage

```
tuik_missing_filters(profile, filters)
```

### Arguments

| | |
|---|---|
| profile | A tuik_flow_profile object. |
| filters | Named list of dimension selections. |

### Value

Character vector containing missing dimension IDs.

---

## tuik_search_concepts *Search concepts used by a TÜİK Dataflow*

---

### Description

Searches component IDs, concept IDs, labels and descriptions in the concepts referenced by a flow profile.

### Usage

```
tuik_search_concepts(
  profile,
  query,
  language = c("both", "tr", "en"),
  limit = 50,
  api_key = NULL
)
```

### Arguments

| | |
|---|---|
| profile | A tuik_flow_profile object. |
| query | Search text. |
| language | One of "both", "tr", or "en". |
| limit | Maximum number of results. Use NULL for all matches. |
| api_key | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |

### Value

A filtered concept tibble.

---

## tuik_search_dataflows *Search TÜİK SDMX Dataflows*

---

### Description

Searches the TÜİK SDMX Dataflow catalogue using a keyword or phrase.

### Usage

```
tuik_search_dataflows(
  query,
  language = c("both", "tr", "en"),
  final_only = FALSE,
  limit = 25,
  fuzzy = TRUE,
  refresh = FALSE,
  dataflows = NULL,
  api_key = NULL
)
```

### Arguments

| | |
|---|---|
| query | Character string containing the search term. |
| language | Which label language should be searched. One of "both", "tr", or "en". Defaults to "both". |
| final_only | Logical. If TRUE, return only final Dataflows. |
| limit | Maximum number of results. Use NULL for all matches. |
| fuzzy | Logical. If TRUE, perform fallback matching when the exact search returns no results. |
| refresh | Logical. If TRUE, refresh the Dataflow catalogue. |
| dataflows | Optional Dataflow catalogue supplied by the user. |
| api_key | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used when retrieving the catalogue. |

## Details

Search fields:

- Dataflow ID
- Turkish Dataflow name
- English Dataflow name
- Referenced DSD ID

Search is case-insensitive and tolerant of Turkish characters.

For example:

"tüketici fiyat" "tuketici fiyat"

are treated equivalently.

If no exact substring match is found, an optional conservative fallback search is performed using shortened word variants. This may help with expressions such as "işsizlik" when catalogue labels contain "işsiz".

## Value

A tibble containing matching Dataflows.

## Examples

```
## Not run:
tuik_search_dataflows("tüketici fiyat")

tuik_search_dataflows("tuketici fiyat")

tuik_search_dataflows("işsizlik")

tuik_search_dataflows(
  "consumer price",
  language = "en"
)

## End(Not run)
```

---

# tuik_validate_filters *Validate SDMX filter values against referenced Codelists*

---

## Description

Checks whether supplied filter values exist in the Codelists referenced by the Dataflow's DSD.

## Usage

```
tuik_validate_filters(profile, filters, stop_on_error = TRUE, api_key = NULL)
```

## Arguments

| | |
|---|---|
| profile | A tuik_flow_profile object. |
| filters | Named list of dimension selections. |
| stop_on_error | Logical. If TRUE, stop when invalid codes are found. |
| api_key | Optional API key. If NULL, the TUIK_SDMX_API_KEY environment variable is used. |

## Value

A tibble with one row per supplied filter value.

# Index

tuik_attributes, 2
tuik_available_codes, 3
tuik_available_codes(), *16*
tuik_build_key, 4
tuik_concepts, 5
tuik_dataflows, 6
tuik_decode_data, 7
tuik_dimension_codes, 9
tuik_dimensions, 10
tuik_explore, 10
tuik_filter_dimensions, 11
tuik_filter_template, 12
tuik_flow_profile, 13
tuik_flow_profile(), *16*
tuik_get_codelist, 14
tuik_get_concept_scheme, 14
tuik_get_data, 15
tuik_get_data(), *8*, *10, 11*
tuik_get_dataflow, 17
tuik_get_dsd, 18
tuik_key_dimensions, 19
tuik_measures, 19
tuik_missing_filters, 20
tuik_search_concepts, 20
tuik_search_dataflows, 21
tuik_validate_filters, 22