Package 'PxWebApiData'

May 7, 2026

Type Package

Title PX-Web Data by API

Version 1.9.0

Date 2026-02-02

Maintainer Øyvind Langsrud <oyl@ssb.no>

Imports httr, rjstat, jsonlite, pxweb

VignetteBuilder knitr

Suggests knitr, rmarkdown, testthat

Description

Function to read PX-Web data into R via API. The example code reads data from the three national statistical institutes, Statistics Norway, Statistics Sweden and Statistics Finland.

License MIT + file LICENSE

Encoding UTF-8

URL https://statisticsnorway.github.io/ssb-pxwebapidata/, https://github.com/statisticsnorway/ssb-pxwebapidata

BugReports https://github.com/statisticsnorway/ssb-pxwebapidata/issues

RoxygenNote 7.3.2

NeedsCompilation no

Author Øyvind Langsrud [aut, cre],
Jan Bruusgaard [aut],
Solveig Bjørkholt [ctb],
Susie Jentoft [ctb]

Repository CRAN

Date/Publication 2026-02-02 10:20:02 UTC

Contents

`ApiData` *PX-Web Data by API*

Description

A function to read PX-Web data into R via API. The example code reads data from the three national statistical institutes, Statistics Norway, Statistics Sweden and Statistics Finland.

Usage

```
ApiData(
  urlToData,
  ...,
  getDataByGET = FALSE,
  returnMetaData = FALSE,
  returnMetaValues = FALSE,
  returnMetaFrames = FALSE,
  returnApiQuery = FALSE,
  defaultJSONquery = c(1, -2, -1),
  verbosePrint = FALSE,
  use_factors = FALSE,
  urlType = "SSB",
  apiPackage = "httr",
  dataPackage = "rjstat",
  returnDataSet = NULL,
  makeNAstatus = TRUE,
  responseFormat = "json-stat2"
)

GetApiData(..., getDataByGET = TRUE)

pxwebData(..., apiPackage = "pxweb", dataPackage = "pxweb")

PxData(..., apiPackage = "pxweb", dataPackage = "rjstat")

ApiData1(..., returnDataSet = 1)
```

```
ApiData2(..., returnDataSet = 2)

ApiData12(..., returnDataSet = 12)

GetApiData1(..., returnDataSet = 1)

GetApiData2(..., returnDataSet = 2)

GetApiData12(..., returnDataSet = 12)

pxwebData1(..., returnDataSet = 1)

pxwebData2(..., returnDataSet = 2)

pxwebData12(..., returnDataSet = 12)

PxData1(..., returnDataSet = 1)

PxData2(..., returnDataSet = 2)

PxData12(..., returnDataSet = 12)
```

Arguments

urlToData	url to data or id of SSB data
...	specification of JSON query for each variable
getDataByGET	When TRUE, readymade dataset by GET
returnMetaData	When TRUE, metadata returned
returnMetaValues	When TRUE, values from metadata returned
returnMetaFrames	When TRUE, values and valueTexts from metadata returned as data frames
returnApiQuery	When TRUE, JSON query returned
defaultJSONquery	specification for variables not included in ...
verbosePrint	When TRUE, printing to console
use_factors	Parameter to fromJSONstat defining whether dimension categories should be factors or character objects.
urlType	Parameter defining how url is constructed from id number. Currently two Statistics Norway possibilities: "SSB" (Norwegian) or "SSBen" (English)
apiPackage	Package used to capture json(-stat) data from API: "httr" (default) or "pxweb"
dataPackage	Package used to transform json(-stat) data to data frame: "rjstat" (default) or "pxweb"
returnDataSet	Possible non-NULL values are 1, 2 and 12. Then a single data set is returned as a data frame.

- 1: The first data set
- 2: The second data set
- 12: Both data sets combined

`makeNAstatus` When TRUE and when dataPackage is "rjstat" and when missing entries in value, the function tries to add an additional variable, named NAstatus, with status codes. An explanation of these status codes is provided in the note part of the comment attribute, i.e. what you get with note(). See the bottom example.

`responseFormat` Response format to be used when apiPackage and dataPackage are defaults ("json-stat" or "json-stat2").

Details

Each variable is specified by using the variable name as input parameter. The value can be specified as: TRUE (all), FALSE (eliminated), imaginary value (top), variable indices, original variable id's (values) or variable labels (valueTexts). Reversed indices can be specified as negative values. Indices outside the range are removed. Variables not specified is set to the value of defaultJSONquery whose default means the first and the two last elements.

The value can also be specified as a (unnamed) two-element list corresponding to the two query elements, filter and values. In addition it possible with a single-element list. Then filter is set to 'all'. See examples.

A comment attribute with the elements label, source, and updated is added to the output as a named character vector. When available, elements originating from tableid, contents, and note are also included, resulting in a vector with at least three elements. Use comment() to view these selected metadata elements. Alternatively, use info() or note() to extract specific parts. The documentation for these two functions provides further details.

Functionality in the package pxweb can be utilized by making use of the parameters apiPackage and dataPackage as implemented as the wrappers PxData and pxwebData. With data sets too large for ordinary downloads, PxData can solve the problem (multiple downloads). When using pxwebData, data will be downloaded in px-json format instead of json-stat and the output data frame will be organized differently (ContentsCode categories as separate variables).

Value

list of two data sets (label and id)

Note

See the package vignette for aggregations using filter agg.

Examples

```
# Note: Example with "readymade datasets" has been removed.
# SSB announced that this service will be discontinued in 2025.
# It is replaced here with an example using PxWebApi v2,
# which supports GET queries and richer options.

url <- "https://data.ssb.no/api/pxwebapi/v2/tables/05810/data?lang=en"
x <- ApiData(url, getDataByGET = TRUE)
```

```
x[[1]]    # The label version of the dataset
x[[2]]    # The id version of the dataset
names(x)
note(x)
info(x)
comment(x)

# As above, but with single dataset output
x1 <- ApiData1(url, getDataByGET = TRUE) # as x[[1]]
x2 <- ApiData2(url, getDataByGET = TRUE) # as x[[2]]
ApiData12(url, getDataByGET = TRUE)      # Combined

# Note: Instead of setting getDataByGET = TRUE manually,
# you can use the wrapper functions GetApiData() or GetApiData12().
# In addition, there are wrapper functions GetApiData1() and GetApiData2(),
# which correspond to ApiData1() and ApiData2().


##### Special output
ApiData("https://data.ssb.no/api/v0/en/table/11419", returnMetaData = TRUE)   # meta data
ApiData("https://data.ssb.no/api/v0/en/table/11419", returnMetaValues = TRUE) # meta data values
ApiData("https://data.ssb.no/api/v0/en/table/11419", returnMetaFrames = TRUE) # list of data frames
ApiData("https://data.ssb.no/api/v0/en/table/11419", returnApiQuery = TRUE)  # query using defaults


##### Ordinary use     (makeNAstatus is in use in first two examples)

# NACE2007 as imaginary value (top 10), ContentsCode as TRUE (all), Tid is default
x <- ApiData("https://data.ssb.no/api/v0/en/table/11419", NACE2007 = 10i, ContentsCode = TRUE)

# Two specified and the last is default (as above) - in Norwegian change en to no in url
x <- ApiData("https://data.ssb.no/api/v0/no/table/11419", NACE2007 = 10i, ContentsCode = TRUE)

# Number of residents (bosatte) last year, each region
x <- ApiData("https://data.ssb.no/api/v0/en/table/04861", Region = TRUE,
        ContentsCode = "Bosatte", Tid = 1i)

# Number of residents (bosatte) each year, total
ApiData("https://data.ssb.no/api/v0/en/table/04861", Region = FALSE,
        ContentsCode = "Bosatte", Tid = TRUE)

# Some years
ApiData("https://data.ssb.no/api/v0/en/table/04861", Region = FALSE,
        ContentsCode = "Bosatte", Tid = c(1, 5, -1))

# Two selected regions
ApiData("https://data.ssb.no/api/v0/en/table/04861", Region = c("1103", "0301"),
        ContentsCode = 2, Tid = c(1, -1))


##### Using id instead of url, unnamed input and verbosePrint
ApiData(4861, c("1103", "0301"), 1, c(1, -1)) # same as below
```

```
ApiData(4861, Region = c("1103", "0301"), ContentsCode=2, Tid=c(1, -1))
names(ApiData(4861,returnMetaFrames = TRUE))  # these names from metadata assumed two lines above
ApiData("4861", c("1103", "0301"), 1, c(1, -1),  urlType="SSBen")
ApiData("01222", c("1103", "0301"), c(4, 9:11), 2i, verbosePrint = TRUE)


##### Advanced use using list. See details above. Try returnApiQuery=TRUE on the same examples.
ApiData(4861, Region = list("03*"), ContentsCode = 1, Tid = 5i) # "all" can be dropped from the list
ApiData(4861, Region = list("all", "03*"), ContentsCode = 1, Tid = 5i)  # same as above
ApiData(04861, Region = list("item", c("1103", "0301")), ContentsCode = 1, Tid = 5i)


##### Using data from SCB to illustrate returnMetaFrames
urlSCB <- "https://api.scb.se/OV0104/v1/doris/sv/ssd/BE/BE0101/BE0101A/BefolkningNy"
mf <- ApiData(urlSCB, returnMetaFrames = TRUE)
names(mf)              # All the variable names
attr(mf, "text")       # Corresponding text information as attribute
mf$ContentsCode        # Data frame for the fifth variable (alternatively  mf[[5]])
attr(mf,"elimination") # Finding variables that can be eliminated
ApiData(urlSCB,        # Eliminating all variables that can be eliminated (line below)
        Region = FALSE, Civilstand = FALSE, Alder = FALSE,  Kon = FALSE,
        ContentsCode  = "BE0101N1", # Selecting a single ContentsCode by text input
        Tid = TRUE)                 # Choosing all possible values of Tid.


##### Using data from Statfi to illustrate use of input by variable labels (valueTexts)
urlStatfi <- "https://pxdata.stat.fi/PXWeb/api/v1/en/StatFin/kuol/statfin_kuol_pxt_12au.px"
ApiData(urlStatfi, returnMetaFrames = TRUE)$Tiedot
ApiData(urlStatfi, Alue = FALSE, Vuosi = TRUE, Tiedot = "Population")  # same as Tiedot = 21


##### Wrappers PxData and pxwebData

# Exact same output as ApiData
PxData(4861, Region = "0301", ContentsCode = TRUE, Tid = c(1, -1))

# Data organized differently
pxwebData(4861, Region = "0301", ContentsCode = TRUE, Tid = c(1, -1))


# Large query. ApiData will not work.
if(FALSE){ # This query is "commented out"
 z <- PxData("https://api.scb.se/OV0104/v1/doris/sv/ssd/BE/BE0101/BE0101A/BefolkningNy",
             Region = TRUE, Civilstand = TRUE, Alder = 1:10, Kon = FALSE,
             ContentsCode = "BE0101N1", Tid = 1:10, verbosePrint = TRUE)
}


##### Small example where makeNAstatus is in use
output <- ApiData("04469", urlType = "SSBen",
                  Tid = "2020", ContentsCode = 1, Alder = TRUE, Region = "3011")
note(output)
```

api_data *PX-Web Data by API*

Description

This function constructs a PxWebApi v2 data URL using `query_url()` and retrieves the data using `get_api_data()`.

Usage

```
api_data(
  url_or_tableid,
  ...,
  url_type = "ssb",
  use_index = FALSE,
  default_query = c(1, -2, -1),
  return_dataset = NULL,
  make_na_status = TRUE,
  verbose_print = FALSE
)

api_data_1(..., return_dataset = 1)

api_data_2(..., return_dataset = 2)

api_data_12(..., return_dataset = 12)
```

Arguments

`url_or_tableid`	A table id, a PxWebApi v2 URL to data or metadata, or metadata returned by `meta_data()` or `meta_frames()`.
`...`	Specification of query for each variable. See 'Details'.
`url_type`	Currently two possibilities: "ssb" (Norwegian) or "ssb_en" (English).
`use_index`	Logical. If TRUE, numeric values are matched against the `index` variable in the metadata, which usually starts at 0. If FALSE (default), numeric values are interpreted as row numbers in the metadata, using standard R indexing. Negative values can be used to specify reversed row numbers.
`default_query`	Specification for variables not included in `...`. The default is `default_query = c(1, -2, -1)`, which selects the first and the two last codes listed in the metadata. Use `default_query = TRUE` and omit specifying individual variables to retrieve entire tables.
`return_dataset`	Possible non-NULL values are `1`, `2` and `12`. Then a single data set is returned as a data frame.

make_na_status	When TRUE and when dataPackage is "rjstat" and when missing entries in value, the function tries to add an additional variable, named NAstatus, with status codes. An explanation of these status codes is provided in the note part of the comment attribute, i.e. what you get with note(). See the bottom example.
verbose_print	When TRUE, printing to console

Details

Each variable is specified by using the variable name as an input parameter.

The value can be specified either as a vector or as a list.

Vector input

When specified as a vector, this results in a valueCodes specification. The vector can be specified in the same way as in a PxWebApi URL. In addition, the specification method inherited from the legacy ApiData() function for PxWebApi v1 can be used:

- TRUE means all values and is equivalent to "*".
- FALSE means eliminated, which is equivalent to removing the variable from the URL. This is meaningful for variables that can be eliminated; see the lower examples in meta_frames().
- Imaginary values represent top, e.g. 3i is equivalent to "top(3)".
- Numeric values are interpreted as row numbers (negative values allowed) or as indices; see the parameter description of use_index.
- Codes can be specified directly, including the use of wildcards such as "*" and "??". Labels may also be used as an alternative to codes.

List input

When the input is a named list, the URL is constructed directly from the names and elements of the list (see examples). It is also possible to omit the name of a list element. In this case, the element results in a valueCodes specification and is processed in the same way as vector input.

Value

A data frame, or a list of data frames, depending on the input and parameters.

Examples

```
obj <- api_data(14162, Region = FALSE, InnKvartering1 = FALSE, Landkoder2 = FALSE,
                ContentsCode = TRUE, Tid = 3i, url_type = "ssb_en")

obj[[1]]    # The label version of the dataset, as returned by api_data_1()
obj[[2]]    # The code version of the dataset, as returned by api_data_2()
names(obj)
info(obj)   # Similar to comment(); see also note() below

 # same as above
 # api_data("https://data.ssb.no/api/pxwebapi/v2/tables/14162/data?lang=en",
 #           default_query = FALSE, ContentsCode = "*", Tid = "top(3)")

 # also same as above
```

```
# api_data(14162, url_type = "ssb_en", default_query = FALSE,
#          ContentsCode = list(valueCodes = "*"),
#          Tid = list(valueCodes = "top(3)"))

api_data_1("https://data.ssb.no/api/pxwebapi/v2/tables/09546/data?lang=en",
           Region = FALSE, SkoleSTR = "07", GrSkolOrgForm = "4",
           EierforhSkole = 1:2, ContentsCode = TRUE, Tid = "202?")

api_data_2("https://data.ssb.no/api/pxwebapi/v2/tables/07459/data?lang=en",
           Region = list(codelist = "agg_KommSummer",
                         valueCodes = c("K-3101", "K-3103"),
                         outputValues = "aggregated"),
           Kjonn = TRUE,
           Alder = list(codelist = "agg_TodeltGrupperingB",
                        valueCodes = c("H17", "H18"),
                        outputValues = "aggregated"),
           ContentsCode = 1,
           Tid = 2i)

# codes and labels can be mixed
api_data_12(4861,
            Region = c("Sarpsborg", "3103", "402?"),
            ContentsCode = "Bosatte",
            Tid = c(1, -1),
            url_type = "ssb_en")


# A Statistics Sweden example
api_data_12("https://statistikdatabasen.scb.se/api/v2/tables/TAB4537/data?lang=en",
            Region = "??",
            Kon = FALSE)


# Use default_query = TRUE to retrieve entire tables
out <- api_data_2("https://data.ssb.no/api/pxwebapi/v2/tables/10172/data?lang=en",
                          default_query = TRUE)
out[14:22, ]  # 9 rows printed


# Use note() for explanation of status codes (see api_data() parameter makeNAstatus)
note(out)

# info() and note() return parts of the comment attribute
info(out)
comment(out)
```

---

get_api_data *PX-Web Data by API (GET)*

Description

A function to read PX-Web data into R via API using GET.

Usage

```
get_api_data(
  url,
  return_dataset = NULL,
  make_na_status = TRUE,
  verbose_print = FALSE,
  use_ensure_json_stat2 = "auto"
)

get_api_data_1(..., return_dataset = 1)

get_api_data_2(..., return_dataset = 2)

get_api_data_12(..., return_dataset = 12)
```

Arguments

`url`	A PxWeb API URL to data.
`return_dataset`	Possible non-NULL values are `1`, `2` and `12`. Then a single data set is returned as a data frame.
`make_na_status`	When TRUE and when dataPackage is `"rjstat"` and when missing entries in `value`, the function tries to add an additional variable, named `NAstatus`, with status codes. An explanation of these status codes is provided in the note part of the comment attribute, i.e. what you get with `note()`. See the bottom example.
`verbose_print`	When TRUE, printing to console
`use_ensure_json_stat2`	`TRUE`, `FALSE`, or `"auto"` (default). If `"auto"`, the URL is modified by `ensure_json_stat2()` only when `"/v2/"` is detected in the URL.
`...`	Additional arguments passed to `get_api_data()`.

Value

A data frame, or a list of data frames, depending on the input and parameters.

Examples

```
url <- paste0(
  "https://data.ssb.no/api/pxwebapi/v2/tables/03013/data?lang=en",
  "&valueCodes[Konsumgrp]=??",
  "&valueCodes[ContentsCode]=KpiIndMnd",
  "&valueCodes[Tid]=top(2)"
)

get_api_data_2(url)
```

info *Extract info or note parts of the comment attribute*

Description

The functions `info()` and `note()` provide access to different parts of an object's comment attribute, which is accessed by `comment()`.

Usage

```
info(x)

note(x)
```

Arguments

`x`	Object with a `comment` attribute.

Details

The `comment` attribute is assumed to be derived from JSON-stat2 metadata, where some elements originate from text in a *note* field.

The `comment` attribute of data downloaded by the package is constructed by `c(unlist(obj[c("label", "source", "updat`
`unlist(obj$extension$px[c("tableid", "contents")]), unlist(obj["note"]))`

where `obj` is a list containing the JSON-stat2 metadata. Thus, possible none-existing elements are ignored.

Value

- `note()` returns all elements in the `comment` attribute that originate from the *note* field. The `"note"` name is then removed.
- `info()` returns the remaining elements in the `comment` attribute.

list_records_to_df *Convert a list of records to a data.frame*

Description

Converts a list where each element represents a record (row) into a `data.frame`. Missing fields are filled with `NA`.

Usage

```
list_records_to_df(x)
```

Arguments

x	A list of records. Each record should be a named list or similar structure. Records may have different sets of fields.

Details

List-valued fields are flattened by collapsing their contents into a single character value. This guarantees that the returned data frame contains no list columns, but nested structure is not preserved.

This function is intended for row-oriented data structures, such as JSON arrays of objects.

Value

A `data.frame` with one row per record and columns corresponding to the union of all field names.

Note

This function is written and documented with help from ChatGPT.

Examples

```
x <- list(
  list(id = 1, name = "Ada", tags = list("a", "b")),
  list(id = 2, name = "Bo"),
  list(id = 3, name = "Cy", tags = list("x"))
)

list_records_to_df(x)
```

`list_to_df_expand` *Create a data.frame from a structured list*

Description

Converts a structured, column-oriented list into a flat `data.frame`, using the first element of the list to define the expected structure (length and names).

Usage

```
list_to_df_expand(x, category_col = "category", dropped_attr = "dropped")
```

Arguments

x	A named list. Each element is expected to have the same length and names as the first element.
category_col	Name of the column containing category labels. If `NULL`, no category column is added and category labels are used as row names instead.
dropped_attr	Name of the attribute used to store excluded elements (stored unchanged). If `NULL`, no such attribute is added.

Details

Only elements matching the structure of the first element are included in the result. Other elements are excluded. Optionally, excluded elements can be stored unchanged as an attribute.

Nested lists or multi-element vectors are expanded into multiple columns so that the returned data frame never contains list columns.

This function is intended for column-oriented or table-like list structures, such as those commonly found in JSON metadata or dimension specifications.

Value

A `data.frame` with one row per category and one or more columns per retained list element.

Note

This function is written and documented with help from ChatGPT.

Examples

```
x <- list(
  A = c(a = 1, b = 2, c = 3),
  B = list(
    a = c(x = 10, y = 20),
    b = c(x = 11, y = 21),
    c = c(x = 12, y = 22)
  ),
  bad = c(a = 1, b = 2)  # wrong length -> excluded
)

df <- list_to_df_expand(x)
df

attr(df, "dropped")

# Use row names instead of a category column:
df2 <- list_to_df_expand(x, category_col = NULL)
df2

# Disable storing excluded elements:
df3 <- list_to_df_expand(x, dropped_attr = NULL)
df3
```

---

meta_code_list *PxWebApi v2 metadata for a code list*

Description

Retrieves metadata for a code list and returns it as an R object.

Usage

```
meta_code_list(url, as_frame = TRUE)
```

Arguments

url	A PxWebApi v2 URL to metadata for a code list.
as_frame	Logical. When TRUE, the metadata is structured as a data frame, with additional information stored in an attribute named "extra".

Value

An R object containing metadata for the code list. When as_frame = TRUE, the result is a data frame.

Examples

```
metaframes <- meta_frames(7459, url_type = "ssb_en")
url <- attr(metaframes[["Region"]], "code_lists")[["links"]][3]
print(url)

df <- meta_code_list(url)

print(df)
print(attr(df, "extra")[1:3])
```

---

meta_data *PxWebApi v2 metadata for a table*

Description

Retrieves metadata for a table using the PxWebApi v2 endpoint and returns it as an R object via jsonlite::read_json().

Usage

```
meta_data(url_or_tableid, url_type = "ssb")
```

Arguments

url_or_tableid	Either a numeric table id or a PxWebApi v2 URL to data. When a data URL is supplied, it is internally converted to a metadata URL.
url_type	Currently two possibilities: "ssb" (Norwegian) or "ssb_en" (English).

Value

A list containing table metadata read by jsonlite::read_json() after internal URL adjustments via ensure_json_stat2(). The data URL is stored in the comment attribute of the returned object.

Examples

```
metadata1 <- meta_data(8991, url_type = "ssb_en")
metadata2 <- meta_data(
  "https://statistikdatabasen.scb.se/api/v2/tables/TAB1525/data?lang=en"
)

print(metadata1[1:4])
print(metadata2[1:4])
```

`meta_frames` *Structured PxWebApi v2 metadata*

Description

Structures selected parts of table metadata into data frames.

Usage

```
meta_frames(url_or_tableid, url_type = "ssb")
```

Arguments

`url_or_tableid`	A table id, a PxWebApi v2 URL to data or metadata, or previously retrieved metadata.
`url_type`	Currently two possibilities: "ssb" (Norwegian) or "ssb_en" (English).

Details

Metadata related to the categories of dimensional variables are returned as data frames, with additional information stored as attributes.

Value

A named list of data frames. Additional metadata is stored as attributes on the data frames. The data URL is stored in the comment attribute of the returned object.

Examples

```
metaframes <- meta_frames(7459, url_type = "ssb_en")

names(metaframes)

metaframes[["ContentsCode"]]
metaframes[["Kjonn"]]

# Extra information stored as an attribute on a data frame
attr(metaframes[["Region"]], "extra")[[1]][[1]]
```

```
# Code list information as a data frame stored as another attribute
attr(metaframes[["Region"]], "code_lists")

# Information about elimination possibilities
attr(metaframes[["Region"]], "elimination")
attr(metaframes[["ContentsCode"]], "elimination")
sapply(metaframes, attr, "elimination") # elimination info for all variables
```

`query_url` *PX-Web API query URL*

Description

PX-Web API query URL

Usage

```
query_url(
  url_or_tableid,
  ...,
  url_type = "ssb",
  use_index = FALSE,
  default_query = c(1, -2, -1)
)
```

Arguments

`url_or_tableid`	A table id, a PxWebApi v2 URL to data or metadata, or metadata returned by `meta_data()` or `meta_frames()`.
`...`	Specification of query for each variable. See 'Details' in `api_data()`.
`url_type`	Currently two possibilities: "ssb" (Norwegian) or "ssb_en" (English).
`use_index`	Logical. If TRUE, numeric values are matched against the `index` variable in the metadata, which usually starts at 0. If FALSE (default), numeric values are interpreted as row numbers in the metadata, using standard R indexing. Negative values can be used to specify reversed row numbers.
`default_query`	Specification for variables not included in `...`. The default is `default_query = c(1, -2, -1)`, which selects the first and the two last codes listed in the metadata. Use `default_query = TRUE` and omit specifying individual variables to retrieve entire tables.

Value

A PxWeb API URL to data, with query parameters added according to the input.

Examples

```
query_url(4861,
          Region = FALSE,
          ContentsCode = "Bosatte",
          Tid = c(1, 5, -1),
          url_type = "ssb_en")


query_url("https://data.ssb.no/api/pxwebapi/v2/tables/08991/data?lang=en",
          Fangst2 = FALSE,
          Elver = FALSE,
          ContentsCode = TRUE,  # same as "*"
          Tid = "top(5)")       # same as 5i



query_url("https://data.ssb.no/api/pxwebapi/v2/tables/07459/data?lang=en",
          Region = FALSE,
          Kjonn = TRUE,
          Alder = list(codelist = "agg_TodeltGrupperingB",
                       valueCodes = c("H17", "H18"),
                       outputValues = "aggregated"),
         ContentsCode = 1,
         Tid = 4i)
```

Index