2024-10-04

Mikael Nordberg
Statistics Sweden

PxWebApi 1.0 description

Content

1 Readers' guide

1.1 What this instruction covers

This manual describes how to use the PxWeb API. The manual is primarily intended for people who want an introduction on how to use the API. The reader is not required to have any type of PxWeb experience to understand the content, but it helps. It also helps to have some knowledge of the HTTP protocol.

Note that the examples in this documentation are fictional and may not work in the real database of Statistics Sweden.

1.2 How this document is organised

Chapters 3 and 4 are intended for the end user who wants to learn more about how to access the API. The other chapters describe the API from a maintenance point of view.

1.3 Terminology

- HTTP verb – Can be looked upon as HTTP request methods where GET and POST are the most commonly used.
- HTTP response code – The HTTP protocol uses different response codes to indicate the response status from the user request. For a list of response codes, see http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html

2 Getting started

If you just wish to use the API and want to familiarize yourself with it, you should read Chapters 3 and 4. If you are interested in enabling the API to expose your PxWeb database, then you should read all the chapters.

3 URLs

The URL is the way to access the API and it has some similarities to a RESTful API.

The URLs are constructed with different components:

```
API-NAME/API-VERSION/LANGUAGE/DATABASE-ID/<LEVELS>/TABLE-ID
```

- API-NAME defines the root part of the API.
- API-VERSION defines the version of the API.
- LANGUAGE defines the language of the data retrieval. DATABASE-ID defines the database where the statistical cubes are stored.
- LEVELS defines zero or more levels that show the various divisions in the database.
- TABLE-ID defines the identity of the table

Example showing the URL for the table BefProgFoddaMedel11:

```
/axis/v1/sv/ssd/BE/BE0401/BE0401B/BefProgFoddaMedel11
```

Suggested URL for implementation at Statistics Sweden would look like:

```
http://api.scb.se/OV0104/v1/doris/en/ssd/BE/BE0401/BE0401B/BefProgFoddaMedel11
```

3.1 API-NAME

Browsing the API-NAME will redirect the caller to an information page about the API or an Http 404 Not Found response.

3.2 API-VERSION

Browsing the /API-NAME/API-VERSION will redirect the user to a page with information, e.g. terms of usage, features, etc., for the specific version of the API.

Notice that currently you will get an Http 404 Not Found response.

3.3 LANGUAGE

Browsing (HTTP verb GET) the /API-NAME/API-VERSION/LANGUAGE will result in a JSON formatted result page listing the available databases for that language.

3.4 DATABASE-ID

Browsing (HTTP verb GET) the /API-NAME/API-VERSION/LANGUAGE/DATABASE-ID will result in a JSON formatted result page listing the first level nodes of the specified database for the specified language.

For the result format, see 4.1.

3.5 LEVEL1…LEVELN

Browsing (HTTP verb GET) the /API-NAME/API-VERSION/LANGUAGE/DATABASE-ID/LEVEL1 to /API-NAME/API-VERSION/LANGUAGE/DATABASE-ID/LEVEL1.../LEVELN will result in a JSON formatted result page listing the available levels and tables for the specified databases for that language and that level.

For the result format, see 4.2.

3.6 TABLE-ID

Browsing (HTTP verb GET) the /API-NAME/API-VERSION/LANGUAGE/DATABASE-ID/LEVEL1.../LEVELN/TABLE-ID will result in a JSON formatted result page specifying the metadata of the table.

For the result format, see 4.3.

Browsing (HTTP verb POST) the /API-NAME/API-VERSION/LANGUAGE/DATABASE-ID/LEVEL1.../LEVELN/TABLE-ID requires a JSON formatted query object. The query object specifies what data should be retrieved from the data cube. The result will be formatted in the format specified in the query.

For the query format, see 4.4. For the JSON formatted response table, see 4.5.

4 JSON formats

4.1 Database result list

This is an array of database objects that has an identity, dbid and a textual description, text. There are three types of node objects: l, t and h, where l is a sublevel, t is a table and h is a heading.

Example result of the URL “/doris/en” shows that we have one database.


```
[{"dbid":"ssd","text":"Statistics Sweden"}]
```


4.2 Database levels result list

This is an array of node objects that has an identity, id and a textual description, text and a type.

Example result of the URL “/doris/en/ssd” shows that we have 15 root nodes in the database.


```
[
   {"id":"BE", "type":"l", "text":"Population"},
   {"id":"FM", "type":"l", "text":"Financial markets"},
   {"id":"HE", "type":"l", "text":"Household finances"},
   {"id":"LE", "type":"l", "text":"Living conditions"},
   {"id":"MI", "type":"l", "text":"Environment"},
   {"id":"NV", "type":"l", "text":"Business activities"},
   {"id":"PR", "type":"l", "text":"Prices and Consumption"},
   {"id":"AM", "type":"l", "text":"Labour market"},
   {"id":"BO", "type":"l", "text":"Housing, construction and building"},
   {"id":"HA", "type":"l", "text":"Trade in goods and services"},
   {"id":"JO", "type":"l", "text":"Agriculture, forestry and fishery"},
   {"id":"ME", "type":"l", "text":"Democracy"},
   {"id":"NR", "type":"l", "text":"National accounts"},
   {"id":"OE", "type":"l", "text":"Public finances"},
   {"id":"UF", "type":"l", "text":"Education and research"}
]
```


Example result of the URL “/doris/en/ssd/BE/BE0401/BE0401B” shows that we have 4 table nodes in the database.


```
[

   {   "id": "BefolkprognRev2009", "type": "t",

       "text": "Population by age and sex. Year 2009-2110 2009 - 2110"},

   {   "id": "BefolkprognRev2010", "type": "t",

       "text": "Population by age and sex. Year 2010-2110 2010 - 2110"},

   {   "id": "BefolkprognRev4", "type": "t",

       "text": "Population size by age and sex. Year 2005-2050 2004 - 2050"},

   {   "id": "BefolkprognRev2008", "type": "t",

       "text": "Population by age and sex. Year 2008-2110 2008 - 2110"}

]
```

A heading node object can look like this:

```
[
    {"id":"XX", "type":"h", "text":"Population heading"}
]
```

4.3 Table metadata result

This has a title property and an array of variables. The variable object has four properties: code, text, elimination and time. The code and text properties are mandatory properties, while the elimination and time are optional. If time or elimination is not specified, the default value of “n” is used. The properties time and elimination could have either y (yes) or n (no) specified. If a variable has elimination set to “y”, one can then omit selecting a value for that variable. There can only be one variable that has the time property set to “t” for a table.

It also contains two lists. One that contains the codes for the all the values of which the variable can assume and one list of all the presentation text for the values.

Example result of the URL “/doris/sv/ssd/BE/BE0401/BE0401B/BefProgFoddaMedel10” shows that the table consist of four dimensions: region, age, gender and period.


```
{

   "title": "Births by country of birth, age and period",
   "variables":
   [

       {   "code": "Fodelseland",
           "text": "country of birth",

           "values":["010","020","030","040","050","060","070"],

           "valueTexts":

           ["Sweden","Nordic countries excl. Sweden",
            "EU excl. Nordic countries",

           "Europe excl. EU and Nordic countries","Low HDI excl. Europe",
           "Medium HDI excl. Europe","High HDI excl. Europe"],
           "elimination": true},

       {   "code": "Alder",

           "text": "age",
           "values":

           ["-14","15","16","17","18","19","20","21","22","23",
           "24","25","26","27","28","29","30","31","32","33",
           "34","35","36","37","38","39","40","41","42","43",
           "44","45","46","47","48","49+"],

           "valueTexts":

           ["-14 years","15 years","16 years","17 years",
            "18 years","19 years","20 years","21 years",
            "22 years","23 years","24 years","25 years",
```

```
            "26 years","27 years","28 years","29 years",
            "30 years","31 years","32 years","33 years",
            "34 years","35 years","36 years","37 years",
            "38 years","39 years","40 years","41 years",
            "42 years","43 years","44 years","45 years",
            "46 years","47 years","48 years","49+ years"],

           "elimination": true},

      {  "code":  "ContentsCode",
           "text": "observations",
           "values":["BE0401M2"],

           "valueTexts":["Births"]},

      {   "code": "Tid",

           "text": "period",
           "values":

           ["2010","2011","2012","2013","2014","2015","2016",
           "2017","2018","2019","2020","2021","2022","2023",
           "2024","2025","2026","2027","2028","2029","2030",
           "2031","2032","2033","2034","2035","2036","2037",
           "2038","2039","2040","2041","2042","2043","2044",
           "2045","2046","2047","2048","2049","2050","2051",
           "2052","2053","2054","2055","2056","2057","2058",
           "2059"],

           "valueTexts":
           ["2010","2011","2012","2013","2014","2015","2016",
           "2017","2018","2019","2020","2021","2022","2023",
           "2024","2025","2026","2027","2028","2029","2030",
           "2031","2032","2033","2034","2035","2036","2037",
           "2038","2039","2040","2041","2042","2043","2044",
           "2045","2046","2047","2048","2049","2050","2051",
           "2052","2053","2054","2055","2056","2057","2058",
           "2059"],
           "time": true

      }

   ]

}
```

4.4 Table retrieval query

The table retrieval query consists of two parts: the actual query and the response object.

The query consists of objects that specify which values are selected for each variable. If a selection is missing for a variable, then the values are selected by the following rules:

1. If the variable has elimination set to “yes” and has an elimination value (a total value), then only that value is selected.
2. If the variable has elimination set to “yes” but has no elimination value, then the aggregated sum of all values is used.
3. If the variable has elimination set to “no”, then all values for that variable are selected.

The selection consists of a property of the variable code and a selection which has a filter and an array of values.

The filter specifies how the values are given. Supported filters are:

- Item. This filter lists valid values in the values collection.
- All. This filter uses a wildcard selector on the values. Each wildcard selector is given in the values collection. Only one wild card is allowed. E.g. 01* gives all values that starts with 01, * gives all values.
- Top. This is used to select the first number of values. The amount of values is given by the first value in the values collection. If the variable is a time variable, then this will select the latest number of time periods.
- Agg. This states that the values listed in the values collection are aggregated. The identity of the aggregation is given after the colon, e.g. agg:ageG5.
- Vs. This states that the values listed in the values collection are from a different value set. The identity of the value set is given after the colon, e.g. vs:regionX.

The response object is optional. If no response object is specified, a px file formatted result will be returned to the client.

Supported formats are:

- px
- csv
- csv2
- csv3

- json
- xlsx
- json-stat
- json-stat2
- sdmx

The format json-stat will return the response as JSON-stat version 1.2 and the format json-stat2 will return the response as JSON-stat 2.0.

Example POST query to the URL "/doris/en/ssd/BE/BE0401/BE0401B/BefProgFoddaMedel10". We specify all values for each variable that we are interested in. Since the gender was eliminable, we did not specify it, so we eliminate it.

```
{

"query": [{"code":"Fodelseland",
           "selection":{"filter":"item", "values":["010","020"]}},

          {"code":"Alder",
           "selection":{"filter":"all","values":["*"]}},

          {"code":"Tid",
           "selection":{ "filter":"top","values":["3"]}}],

"response": {"format":"csv"}

}
```

4.4.1 Example of retrivals in format csv

First line contains variable name and content.
Plain text and values on following lines.

```
"region","marital status","age","sex","Population 2023"
"01 Stockholm county","single","20-29 years","men",142540
"01 Stockholm county","single","20-29 years","women",127966
"01 Stockholm county","single","30-39 years","men",127966
"01 Stockholm county","single","30-39 years","women",104126
"01 Stockholm county","single","40-49 years","men",63877
"01 Stockholm county","single","40-49 years","women",50624
"01 Stockholm county","single","50-59 years","men",45647
"01 Stockholm county","single","50-59 years","women",39319
```

4.4.2 Example of retrivals in format csv2

The table is pivoted such that all variables are in the heading. All variabels are placed on the first line and further every value is placed on its own line.

```
"region","marital status","age","sex","year","observations","Population"
"01 Stockholm county","single","20-29 years","men","2023","Population",142540
"01 Stockholm county","single","20-29 years","women","2023","Population",127966
"01 Stockholm county","single","30-39 years","men","2023","Population",127966
"01 Stockholm county","single","30-39 years","women","2023","Population",104126
"01 Stockholm county","single","40-49 years","men","2023","Population",63877
"01 Stockholm county","single","40-49 years","women","2023","Population",50624
"01 Stockholm county","single","50-59 years","men","2023","Population",45647
"01 Stockholm county","single","50-59 years","women","2023","Population",39319
```

4.4.3 Example of retrivals in format csv

This format display variable name and codes instead of description and text.
Changed header on the last colum to show ID for the table instead of text.

```
"Region","Civilstand","Alder","Kon","Tid","ContentsCode","TAB638"
"01","OG","20-29","1","2023","BE0101N1",142540
"01","OG","20-29","2","2023","BE0101N1",127966
"01","OG","30-39","1","2023","BE0101N1",127966
"01","OG","30-39","2","2023","BE0101N1",104126
"01","OG","40-49","1","2023","BE0101N1",63877
"01","OG","40-49","2","2023","BE0101N1",50624
"01","OG","50-59","1","2023","BE0101N1",45647
"01","OG","50-59","2","2023","BE0101N1",39319
```

4.5 Table response

The table response is divided into a metadata and a data part. The metadata part consists of a columns collection that lists the different dimensions and measures of the data cube. It also has a comments collection that specifies comments to a specific value. The data part of the response stores the measures of the data cube.

4.5.1 Metadata part

The column description consists of five properties:

- Code. This is the identifier.
- Text. A textual display name for the column.
- Type. The type of column. This has three possible values.
    - *d* - a dimension. This is the default value. If it has not been set, it is assumed to be a dimension.
    - *t* - this is the time dimension
    - c - this is a measure column
- Unit. If it is a “c” column, the unit can be specified telling what unit the measures are specified in. This property should be ignored if it is not a “c” column
- Comment. This is an optional comment for the column/dimension.

4.5.2 Data part

The data part is given in the form of key and values. The key specifies the dimensions and the values are the different measures for the dimensions. The order of how the different values are specified in the key should be the same order as they are specified in the columns array. There is also an optional comments array for each key-values pair that specifies a comment for a measure. The index of the comment corresponds to the same index of the measure in the values array.


```
"columns":[{"code":"region", "text":"Region"},
           {"code":"ageG5", "text":"Age",
           "comment":"Citizens under the age of 6 are not…"},
           {"code":"period", "text":"Time", "type":"t"},
           {"code":"x", "text":"Population", "type":"c", "unit":"amount"}],
"comments":[{"variable":"period", "value":"2005",
            "comment":"Preliminary firgues"}]

"data":[{"key":["02","0-7","2003"], "values":[100]},
        {"key":["02","0-7","2004"], "values":[101]},
        {"key":["02","0-7","2005"], "values":[102]},
        {"key":["02","8-14","2003"], "values":[103]},
        {"key":["02","8-14","2004"], "values":[104]},
        {"key":["02","8-14","2005"], "values":[105]},
        {"key":["02","15-23","2003"], "values":[106]},
        {"key":["02","15-23","2004"], "values":[107]},
        {"key":["02","15-23","2005"], "values":[108]},
        {"key":["02","24-33","2003"], "values":[109]},
        {"key":["02","24-33","2004"], "values":[110]},
        {"key":["02","24-33","2005"], "values":[111]},
        {"key":["02","34-54","2003"], "values":[112]},
        {"key":["02","34-54","2004"], "values":[113]},
        {"key":["02","34-54","2005"], "values":[114]},
        {"key":["02","55+","2003"], "values":[115]},
        {"key":["02","55+","2004"], "values":[116]},
        {"key":["02","55+","2005"], "values":[117]},
        {"key":["06","0-7","2003"], "values":[118]},
        {"key":["06","0-7","2004"], "values":[119]},
        {"key":["06","0-7","2005""], "values":[120]},
        {"key":["06","8-14","2003"], "values":[121]},
        {"key":["06","8-14","2004"], "values":[122]},
        {"key":["06","8-14","2005"], "values":[123]},
        {"key":["06","15-23",2003"], "values":[124]},
        {"key":["06","15-23",2004"], "values":[125]},
        {"key":["06","15-23",2005"], "values":[126]},
        {"key":["06","24-33",2003"], "values":[127]},
        {"key":["06","24-33",2004"], "values":[128]},
        {"key":["06","24-33",2005"], "values":[129]},
        {"key":["06","34-54",2003"], "values":[130]},
        {"key":["06","34-54",2004"], "values":[131]},
        {"key":["06","34-54",2005"], "values":[132]},
        {"key":["06","55+","2003"], "values":[133]},
        {"key":["06","55+","2004"], "values":[134], "comment":["Imputated"]},
        {"key":["06","55+","2005"], "values":[135]}],

}
```

5 Limitations

The usage of the API can be limited or not. The limitation is based on the caller IP-address. The way it works is that the caller can make 10 requests during a time window of 10 seconds (these values can be configured later). The time period 10 seconds is a sliding time window. If the caller tries to make more requests than allowed, the API will return a *HTTP 429 Too Many Requests* response.

6 Logging

The log4net framework is used for the usage logging. The API uses the *api- usage* logger to log the data retrievals. Information that is logged is the IP- address of the caller, the statistical table and the number of measures that were retrieved.