

Would it be possible to Access SDMX data provided by External data providers like the OECD, IMF, Eurostat, ECB

There are 3 methods to read SDMX Queries with SAS.

- 1) Use the ISTAT SDMX.jar tool to generate %gettimeseries() macro calls
- 2) Read SDMX Queries with SAS ETS Engines SASEOECD, SASEFRED and SASEQUAN
- 3) Read SDMX Queries using XMLV2 libname engine

Generic example using SDMX queries:

ISTAT provides a SDMX package to use together with Statistical software packages to read SDMX data. <https://github.com/amattioc/SDMX>

SDMX Providers	• European Central Bank	• International Labour Organization
	• BIS	• World Integrated Trade Services
	• Eurostat	• INSEE
	• IMF	• ISTAT
	• World Bank	• Australian Bureau of Statistics
	• UN Data	• National Bank of Belgium
	• OECD	• INEGI

The latest release of packages can be found here:

<https://github.com/amattioc/SDMX/releases/latest>

Run following command:

```
"C:\Program Files\SASHome\SASPrivateJavaRuntimeEnvironment\9.4\jre\bin\java" -classpath  
c:\workshop\viyawhatsnew\sdmx_query\SDMX.jar it.bancaditalia.oss.sdmx.helper.SDMXHelper
```

Downloading the SDMX.jar package for use in SAS and to prepare the queries that select the time series using the gettimeseries.sas macro.

NEW: The SDMX helper can be used from within any statistical tool or as a standalone application (just double click the SDMX.jar)

```
/*Add the downloaded jar file to the classpath of your sas session*/
```

```
/*in SASV9.CFG use: */
```

```
/*-SET CLASSPATH "&path/SDMX/lib/SDMX.jar"; */
```

```
*options set=CLASSPATH "&path/SDMX/lib/SDMX.jar";
```

You can select the Data Provider (e.g. ECB), then select the Data Flow (for Example Exchange Rates EXR) and the Dimensions of the Time Series like Currency=USD.

SDMX Helper Tool

Providers
Actions
Help

Query

Result query: EUROSTAT
NAMA_10_GDP/...

Dataflow selection

Filter flows by name or description:

☐ Regular expression
☐ Case sensitive
☐ Whole word
☐ Search Dataflow Code
☐ Search DSD Code
☐ Search Description
☒ Search all fields

Dataflow	Version	DSD	DSD Ver...	Agency	Description
NAMA_10_GDP	1.0	NAMA_10_GDP	47.0	ESTAT	GDP and main components (output, expenditure and income)
NAMA_10_IP_A21	1.0	NAMA_10_IP_A21	39.1	ESTAT	Labour productivity and unit labour costs at industry level
NAMA_10_IP_LUC	1.0	NAMA_10_IP_LUC	50.0	ESTAT	Labour productivity and unit labour costs
NAMA_10_NFA_BS	1.0	NAMA_10_NFA_BS	41.0	ESTAT	Balance sheets for non-financial assets
NAMA_10_NFA_FL	1.0	NAMA_10_NFA_FL	41.1	ESTAT	Cross-classification of gross fixed capital formation by industry and by asset (flows)
NAMA_10_NFA_ST	1.0	NAMA_10_NFA_ST	47.1	ESTAT	Cross-classification of fixed assets by industry and by asset (stocks)
NAMA_10_NFA_ST&DV_647	1.0	NAMA_10_NFA_ST	47.1	ESTAT	Cross-classification of fixed assets by industry and by asset (stocks)

Dimension selection

Dimension to edit:
Clear selected dimension

Dimension	Description
0/freq	Time frequency
1/unit	Unit of measure
2/na_item	National accounts indicator (ESA 2010)
3/geo	Geopolitical entity (reporting)

Code list selection

Filter codes:
☐ Regular expression
☐ Case sensitive
☐ Whole word
☐ Code
☐ Description
☒ Search Both

Code ID	Code Description
---------	------------------

80* INFO [i.b.o.s.client.RestSdmxClient:runQuery] Contacting web service with query: <https://ec.europa.eu/eurostat/api/dissemination/sdmx/2.1/dataflow/all/all/latest>

80* INFO [i.b.o.s.client.RestSdmxClient:runQuery] Contacting web service with query: https://ec.europa.eu/eurostat/api/dissemination/sdmx/2.1/datastructure/ESTAT/ENPE_NAMA_10_GDP/22.0?references=children

80* INFO [i.b.o.s.client.RestSdmxClient:runQuery] Contacting web service with query: https://ec.europa.eu/eurostat/api/dissemination/sdmx/2.1/datastructure/ESTAT/NAMA_10_GDP/47.0?references=children

This menu option allows you to change the Providers:

SDMX Helper Tool

Providers
Actions
Help

[SDMX_V2] ABS: Australian Bureau of Statistics - SDMX 2.1

[SDMX_V2] BBK: Deutsche Bundesbank

[SDMX_V2] BIS_PUBLIC: Bank for International Settlements

[SDMX_V2] ECB: European Central Bank

• [SDMX_V2] EUROSTAT: Eurostat

[SDMX_V2] EUROSTAT_COMEXT: Eurostat - COMEXT

[SDMX_V2] EUROSTAT_COMP: Eurostat - DG COMP

[SDMX_V2] EUROSTAT_EMPL: Eurostat - DG EMPL

[SDMX_V2] EUROSTAT_GROW: Eurostat - DG GROW

[SDMX_V2] ILO: International Labour Organization

[SDMX_V2] IMF2: New International Monetary Fund endpoint

[SDMX_V2] INEGI: Instituto Nacional de Estadística y Geografía

[SDMX_V2] INSEE: National Institute of Statistics and Economic Studies

[SDMX_V2] ISTAT: Italian National Institute of Statistics

[SDMX_V2] ISTAT_CENSUS_AGR: ISTAT - Agricultural census 2010

[SDMX_V2] ISTAT_CENSUS_IND: ISTAT - Industry and services census 2011

[SDMX_V2] ISTAT_CENSUS_POP: ISTAT - Population and housing census 2011

[SDMX_V2] ISTAT_RI: Italian National Institute of Statistics

[SDMX_V2] NBB: National Bank Belgium

[SDMX_V2] OECD: The Organisation for Economic Co-operation and Development

[SDMX_V2] OECD_REST: The Organisation for Economic Co-operation and Development, RESTRICTED ACCESS

[SDMX_V2] StatsEE: Statistics Estonia (BETA)

[SDMX_V2] UNDATA: Data access system to UN databases

[SDMX_V2] UNICEF: UNICEF

[SDMX_V2] WB: World Bank - World Development Indicators

[SDMX_V2] WITS: World Integrated Trade Solutions

You select the required dimensions of the Query:

For example, Frequency: Annual, Unit of Measure: CLV10_MEUR and geo: BE+FR

The screenshot shows the SDMX Helper Tool interface. The 'Query' section displays the result query: `EUROSTAT NAMA_10_GDP/A.CLV10_MEUR.B1GQ.FR+BE`. The 'Dataflow selection' section shows a table of dataflows with 'nama_10_gdp' selected. The 'Dimension selection' section shows dimensions for frequency, unit, and geo, with 'geo' selected. The 'Codelist selection' section shows the selection of 'FR' for France. The 'Commands in statistical tools' window shows the generated commands for R, MATLAB, and SAS.

Dataflow	Version	DSD	DSD Ve...	Agency	Description
ENPE_NAMA_10_GDP	1.0	ENPE_NAMA_10_GDP	22.0	ESTAT	Gross domestic product at market prices
ENPE_NAMA_10_GDP_EA	1.0	ENPE_NAMA_10_GDP_EA	29.0	ESTAT	Gross domestic product at market prices by expenditure
ENPS_NAMA_10_GDP	1.0	ENPS_NAMA_10_GDP	22.0	ESTAT	Gross domestic product at market prices
ENPS_NAMA_10_GDP_EA	1.0	ENPS_NAMA_10_GDP_EA	29.0	ESTAT	Gross domestic product at market prices by expenditure
NAMA_10_GDP	1.0	NAMA_10_GDP	47.0	ESTAT	GDP and main components (output, expenditure and income)

```

R: result <- getTimeSeries('EUROSTAT', 'NAMA_10_GDP/A.CLV10_MEUR.B1GQ.FR+BE');
MATLAB: result = getTimeSeries('EUROSTAT', 'NAMA_10_GDP/A.CLV10_MEUR.B1GQ.FR+BE');
SAS: %gettimeseries(provider="EUROSTAT", tsKey="NAMA_10_GDP/A.CLV10_MEUR.B1GQ.FR+BE", metadata=1);
  
```

With the Actions menu, you can then convert the selected query to a command:

The screenshot shows the SDMX Helper Tool interface with the 'Actions' menu open. The menu options are 'Copy selection', 'Build commands', and 'Add provider...'. The 'Build commands' option is highlighted.

The command in SAS uses the %gettimeseries () macro call:

The screenshot shows the 'Commands in statistical tools' window. The SAS command is highlighted:

```

SAS: %gettimeseries(provider="EUROSTAT", tsKey="NAMA_10_GDP/...", metadata=1);
  
```

The gettimeseries() macro can be found in the SDMX repository:

01_clonesdmx_repository.sas

02_dynamically_change_classpath.sas

03_gettimeseries.sas macro call

```
/*create the SDMX folder in the home directory of the user*/
```

```
%let homedir=%sysget(HOME);
```

```
%let path=&homedir;
```

```
options dlcreatedir;
```

```
libname data "&path/SDMX";
```

```
/*clone the SDMX git repository to a local folder*/
```

```
data _null_;
```

```
RC = GITFN_CLONE("https://github.com/amattioc/SDMX.git",
```

```
"&path/SDMX");
```

```
run;
```

```
/* example Getting Exchange Rates from ECB data provider*/
```

```
%gettimeseries(provider="ECB", tsKey="EXR.A.USD.EUR.SP00.A", metadata=1);
```

```
%macro
```

```
copytimeseries(tskey=NAMA_10_GDP/A.CLV_I10.B1GQ,outlib=work,metadata=1);
```

```
%let seriesname=%sysfunc(compress(%scan(&tskey,1,%str(+)),.));
```

```
data &outlib..&seriesname;
```

```
set sdmxdata;
```

```
run;
```

```
%if &metadata=1 %then %do;
```

```
data &outlib..&seriesname.meta;
```

```
set sdmxmetadata;
```

```
run;
```

```
data &outlib..&seriesname.ometadata;
```

```
set sdmxobservationsmetadata;
```

```
run;
```

```
%end;
```

```
%mend;
```

```
/* example Getting Exchange Rates from ECB data provider*/
```

```
%gettimeseries(provider="ECB", tsKey="EXR.A.USD.EUR.SP00.A", metadata=1);
```

```

/* example getting Eurostat */

%gettimeseries(provider="EUROSTAT",

tsKey="NAMA_10_GDP/A.CLV_I10.B1GQ.EU27_2020+EU28+EU15+EA+EA20+EA19+EA12+BE+
BG+CZ+DK+DE+EE+IE+EL+ES+FR+HR+IT+CY+LV+LT+LU+HU+MT+NL+AT+PL+PT+RO+SI+SK+FI+
SE+IS+LI+NO+CH+UK+BA+ME+MK+AL+RS+TR+XK",

metadata=1);

/*copytimeseries to outlib=work*/

%copytimeseries(tskey=NAMA_10_GDP/A.CLV_I10.B1GQ,outlib=work,metadata=1);


%gettimeseries(provider="EUROSTAT",

tsKey="NAMA_10_GDP/A.CP_MEUR.B1GQ.EU27_2020+EU28+EU15+EA+EA20+EA19+EA12+BE+
BG+CZ+DK+DE+EE+IE+EL+ES+FR+HR+IT+CY+LV+LT+LU+HU+MT+NL+AT+PL+PT+RO+SI+SK+FI+
SE+IS+LI+NO+CH+UK+BA+ME+MK+AL+RS+TR+XK",

metadata=1);

%copytimeseries(tskey=NAMA_10_GDP/A.CP_MEUR.B1GQ,outlib=work,metadata=1);

%gettimeseries(provider="EUROSTAT",

tsKey="NAMA_10_GDP/A.CLV10_MEUR.B1GQ.EU27_2020+EU28+EU15+EA+EA20+EA19+EA12+
BE+BG+CZ+DK+DE+EE+IE+EL+ES+FR+HR+IT+CY+LV+LT+LU+HU+MT+NL+AT+PL+PT+RO+SI+SK+
FI+SE+IS+LI+NO+CH+UK+BA+ME+MK+AL+RS+TR+XK",

metadata=1);

%copytimeseries(tskey=NAMA_10_GDP/A.CLV10_MEUR.B1GQ,outlib=work,metadata=1)
;

```

Retrieving SDMX Data using SAS ETS engines: (see code below for examples)

(1) Eurostat is available in JSON format using root url: [REST SDMX 2.1 - SDMX Web Services - Eurostat \(europa.eu\)](https://ec.europa.eu/eurostat/web/rest-services)

Example query: [https://ec.europa.eu/eurostat/SDMX/diss-web/rest/data/nama_10_gdp/.CLV10_MEUR.B1GQ.BE/?startperiod=2005&endPeriod=2011](https://ec.europa.eu/eurostat/web/rest-services/getting-started/rest-request)

The data can be mapped and read into SAS using the JSON libname engine.

See "<https://ec.europa.eu/eurostat/web/json-and-unicode-web-services/getting-started/rest-request>" for details.

(2) IMF data (14,742 series) can be read using SASEFRED engine (ETS) see FRED website here: "<https://fred.stlouisfed.org/tags/series?t=imf>"

[The SASEFRED Interface Engine](https://fred.stlouisfed.org/tags/series?t=imf)

(3) ECB data (13 series) can be read using SASEFRED engine- see FRED website here: "<https://fred.stlouisfed.org/tags/series?t=ecb>"

(4) ECB data (212,000 series) can be read using SASEQUAN (another ETS/Econometrics engine). Previously known as QUANDL api, it is now the NASDAQ api , see the following documentation page:

"<https://data.nasdaq.com/data/ECB-european-central-bank/documentation>" for details.

You can get a list of the time series using the link shown on the doc page.

[SAS Help Center: Getting Started: SASEQUAN Interface Engine](#)

Retrieving OECD data SDMX data

https://documentation.sas.com/doc/en/etsug/15.2/etsug_saseoecd_examples01.htm

Example 52.1 Retrieving OECD Gross Domestic Product Data for One Region
(View the complete [code for this example](#).)

You can start building an OECD query for this example on the web page at the following URL:

http://stats.oecd.org/index.aspx?datasetcode=SNA_TABLE1_SNA93

Select **Customize** → **Selection**, which shows the dimension values that are the key values for **Country**, **Transaction**, and **Measure**. Select **Euro area (17 countries)** from the **Country** list. Select **Gross domestic product (output approach)** from the **Transaction** box, and **Current prices** from the **Measure** list. Specify the Observation period to limit the time range to the span 1995 to 2013. On the **Export** tab, select **Developer API**. Then click **Generate API queries**.

The **Data query** box shows the URL for the key values that you selected for **Country**, **Transaction**, and **Measure**:

http://stats.oecd.org/sdmx-json/data/SNA_TABLE1_SNA93/EA17.B1_GA.C/all?startTime=1995&endTime=2013

In your SAS code, use SETID=SNA_TABLE1_SNA93 to indicate the OECD data set. Next, you can specify the INSET n = options by using $n=0,1,2$ for **Country**, **Transaction**, and **Measure**, respectively. The SAS code is shown after the next paragraph, followed by the output, which is shown in [Output 52.1.1](#).

The SET statement reads observations from the input data set *myLib.GSTART* and stores them in a SAS data set named *myGDP*. When you specify the INSET n = option, you name the SAS input data set for each of the n keysets that define your selection of data. The SASEOECD engine takes the crossproduct of all the insets and creates a temporary data set named *CrossKey*. Each row in *CrossKey* defines a unique time series request. Not every row in *CrossKey* yields meaningful data. Only the rows that contain valid data are placed in a JSON file. When a request for data (using the values in each row) generates a valid JSON file, the file is named by concatenating the OUT= option name to the observation number (n) in the *CrossKey* data set that corresponds to the row whose values generated the request. When all the data are retrieved, they are placed in a SAS data set that is named by the OUT= option and that is located in the folder specified by the *physical-name* in the LIBNAME *libref* SASEOECD statement.

```
/*
http://stats.oecd.org/sdmx-
json/data/SNA_TABLE1_SNA93/EA17.B1_GA.C/all?startTime=1995&endTime=2
013
*/
options validvarname=any dlcreatedir;
%let homedir=%sysget(HOME);
%let path=&homedir/viyawhatsnew;

libname oecddata base "&path/sdmx_query/oecddata";

data keylist0;
    length key0 $8;
    key0='EA17'; output; /* country is euro area; 17 countries */
run;

data keylist1;
    length key1 $8;
    key1='B1_GA'; output; /* transaction is GDP; output approach */
run;

data keylist2;
    length key2 $2;
    key2='C'; output; /* measure is current prices */
run;

title 'Request GDP for EA_17 in Current Prices';
LIBNAME myLib saseoecd "&path/sdmx_query/oecddata"
    setid=SNA_TABLE1_SNA93
    inset0=keylist0
    inset1=keylist1
```

```

inset2=keylist2
out=gstart
;

data myGDP;
    set myLib.gstart ;
run;
proc print data=myGDP; run;

```

FMI or IMF Data:

[IMF SDMX Central](#)

Reading SDMX using REST API and XMLV2 engine:

/*Using OECD XML Rest Api: */

```

*filename oecddata url
'https://stats.oecd.org/restsdmx/sdmx.ashx/GetData/QNA/AUS+AUT.GDP+B1_GE.CUR+VOBARSA.Q/
all?startTime=2009-Q2&endTime=2011-Q4&format=compact_v2';

%let path = &path/sdmx_query;

filename map "&path.map.txt";

filename resp "&path.resp.txt";

proc http

URL="https://stats.oecd.org/restsdmx/sdmx.ashx/GetData/QNA/AUS+AUT.GDP+B1_GE.CUR+VOBAR
SA.Q/all?startTime=2009-Q2&endTime=2011-Q4&format=compact_v2"

METHOD="GET"

OUT=resp;

run;quit;

libname resp XMLv2 automap=REPLACE xmlmap=map;

proc datasets;

copy out=WORK in=resp;

run;quit;

proc sql;

%if %sysfunc(exist(WORK.QUERY_FOR_OBS1)) %then %do;

```



```

drop table WORK.QUERY_FOR_OBS1;

%end;

%if %sysfunc(exist(WORK.QUERY_FOR_OBS1,VIEW)) %then %do;

    drop view WORK.QUERY_FOR_OBS1;

%end;

quit;

;

PROC SQL;

    CREATE TABLE WORK.QUERY_FOR_OBS1 AS

        SELECT

            (t2.DataSet_ORDINAL),

            (t2.Series_ORDINAL),

            (t2.Series_LOCATION),

            (t2.Series_SUBJECT),

            (t2.Series_MEASURE),

            (t2.Series_FREQUENCY),

            (t2.Series_TIME_FORMAT),

            (t2.Series_UNIT),

            (t2.Series_POWERCODE),

            (t2.Series_REFERENCEPERIOD),

            (t1.Series_ORDINAL) AS Series_ORDINAL1,

            (t1.Obs_ORDINAL),

            (t1.Obs_TIME),

            (t1.Obs_OBS_VALUE),

            (t5.CompactData_ORDINAL),

            (t5.Header_ORDINAL),

            (t5.ID),

            (t5.Test),

            (t5.Truncated),

            (t5.'Prepared'n) FORMAT=IS8601DT19. INFORMAT=IS8601DT19.,

            (t6.Header_ORDINAL) AS Header_ORDINAL1,

```

```

        (t6.Sender_ORDINAL),
        (t6.Sender_id),
        (t7.DataSet_ORDINAL) AS DataSet_ORDINAL1,
        (t7.Annotations_ORDINAL),
        (t8.Annotations_ORDINAL) AS Annotations_ORDINAL1,
        (t8.Annotation_ORDINAL),
        (t8.AnnotationTitle),
        (t8.AnnotationURL)
FROM
        WORK.OBS t1
        INNER JOIN WORK.SERIES t2 ON (t1.Series_ORDINAL =
t2.Series_ORDINAL)
        INNER JOIN WORK.DATASET t3 ON (t2.DataSet_ORDINAL =
t3.DataSet_ORDINAL)
        INNER JOIN WORK.COMPACTDATA t4 ON (t3.CompactData_ORDINAL
= t4.CompactData_ORDINAL)
        INNER JOIN WORK.'HEADER'n t5 ON (t3.CompactData_ORDINAL =
t5.CompactData_ORDINAL)
        INNER JOIN WORK.SENDER t6 ON (t5.Header_ORDINAL =
t6.Header_ORDINAL)
        INNER JOIN WORK.ANNOTATIONS t7 ON (t3.DataSet_ORDINAL =
t7.DataSet_ORDINAL)
        INNER JOIN WORK.ANNOTATION t8 ON (t7.Annotations_ORDINAL =
t8.Annotations_ORDINAL)
;
QUIT;
RUN;

```

Some documentation:

[SDMX – Statistical Data and Metadata eXchange | Welcome to the SDMX website](#)

[SDMX Connectors for Statistical Software | SDMX – Statistical Data and Metadata eXchange](#)

[.Stat Suite documentation \(sis-cc.gitlab.io\)](#)

Example to use SASEQUAN engine :

<https://sdw-wsrest.ecb.europa.eu/service/data/EXR/M.USD.EUR.SP00.A>

```
/*Example to download ECB data using Nasdag API: */
title 'Germany , Current prices, National currency,
Index';
options validvarname=any;
%let path=c:\workshop\eurostat;

libname mylib "&path";

libname myQ3 sasequan "&path"
    OUTXML=fred3
    AUTOMAP=replace
    MAPREF=MyMap
    XMLMAP="&path\fred3.map"
    APIKEY='Ya9rzjEbW66K1mHphsvb'

IDLIST='ECB/ESA_A_DE_N_1000_COMPHW_0000_YD_D_V_N_I'
    FORMAT=xml
    START='2002-06-30'
    END='2021-12-31'
    /*FREQ='monthly'
    collapse='annual'*/
;

data mylib.thrall;
    set myQ3.fred3;
    label 'Pure Number'n = "Germany , Current prices,
National currency, Index";

run;
proc contents data=mylib.thrall; run;
proc print data=mylib.thrall label; run;
```

Example of Timeseries from IMF using SASEFRED ETS Engine

```
title 'Retrieve Balance of Payment Data for the  
Exports and Imports';  
libname _all_ clear;  
%let path=c:\workshop\eurostat;  
libname fred sasefred "&path"  
OUTXML=fredex01  
AUTOMAP=replace  
MAPREF=MyMap  
XMLMAP="&path\fredex01.map"  
APIKEY='399eb04a24a59583574beea2248db31'  
IDLIST='bopxgs,bopmgs'  
START='1997-01-01'  
END='2011-01-01'  
FREQ='a'  
OUTPUT=1  
AGG='avg'  
FORMAT=xml;  
data export_import;  
set fred.fredex01 ;  
run;  
proc contents data=export_import; run;  
proc print data=export_import; run;
```

Example: Global Price of Natural GAS, EU:

```
/*https://fred.stlouisfed.org/series/PNGASEUUSDM*/  
title 'Retrieve Global Price of Natural GAS, EU in  
USD per Million Metric British';  
libname _all_ clear;  
%let path=c:\workshop\eurostat;  
libname fred sasefred "&path"  
OUTXML=fredex01  
AUTOMAP=replace  
MAPREF=MyMap  
XMLMAP="&path\fredex01.map"  
APIKEY='399eb04a24a59583574beea2248db31'  
IDLIST='PNGASEUUSDM'  
START='1997-01-01'  
END='2022-06-30'  
FREQ='m'  
OUTPUT=1  
AGG='avg'
```

```

FORMAT=xml;
data PNGASEUUSDM;
set fred.fredex01 ;
run;
proc contents data=PNGASEUUSDM; run;
proc print data=PNGASEUUSDM; run;
proc sgplot data=PNGASEUUSDM;
vline date /response=PNGASEUUSDM stat=mean ;
xaxis fitpolicy=rotatethin grid;
yaxis grid;
title "Global Price of Natural Gas, EU";
format date yymm5.;
label pngaseuusdm ="US Dollar per Mill. Metric
British Thermal Unit"
date ="Source: International Monetary Fund";
quit;

/*Example Eurostat SDMX data: */
/*old url:
https://ec.europa.eu/eurostat/SDMX/diss-
web/rest/data/nama\_10\_gdp/.CLV10 MEUR.B1GQ.BE/?startp
eriod=2005&endPeriod=2011
new url:
https://ec.europa.eu/eurostat/api/dissemination/sdmx/
2.1/data/NAMA_10_GDP/?format=sdmx_2.1_generic
*/
%let path=c:\workshop\sdmx;
%let path = &path;

filename map "&path.map.txt";
filename resp "&path.resp.txt";
proc http

URL="https://ec.europa.eu/eurostat/api/dissemination/
sdmx/2.1/data/NAMA_10_GDP/?format=sdmx_2.1_generic"
METHOD="GET"
OUT=resp;
run;quit;

libname resp XMLv2 automap=REPLACE xmlmap=map;

proc datasets;
copy out=WORK in=resp;

```

```
run;quit;
```

```
proc sql;
```

```
create table sdmxrequest as
```

```
select *
```

```
from obs t1, obsdimension t2,  
      obsvalue t3
```

```
where obs.obs_ordinal=obsdimension.obs_ordinal and  
      obs.obs_ordinal=obsvalue.obs_ordinal
```

```
;
```

```
quit;
```