How to API in R

27/01/26

General

Preamble

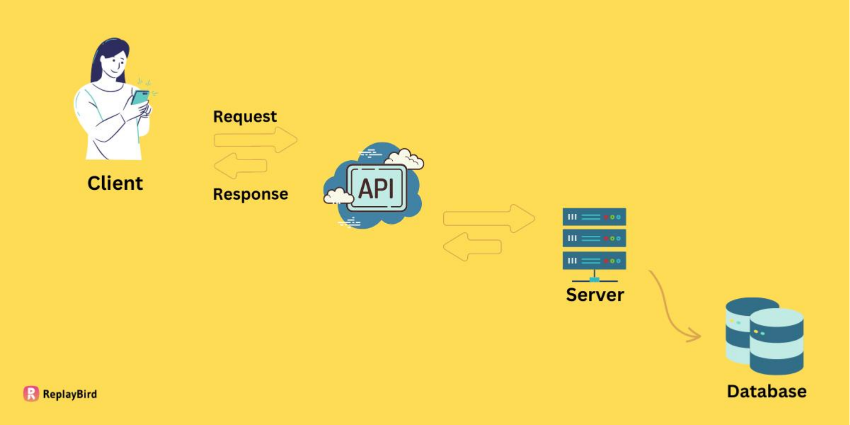
- I cant answer everything, most of it is learning by doing
- Most of the APIs are not really user-friendly documented
- Hopefully it will not take 2 hours

What is an API?

API = Application Programming Interface

- programmatic way to retrieve data from online servers
- can be anything (uploading, downloading, instructing)
- different API types: REST, SDMX, JSON, ... (*think dialects*)

Goal: Format Request in specific way to get specific data from the server



What is SDMX?

The official site for the SDMX community

A global initiative to improve Statistical Data and Metadata eXchange

an API for statistical agencies

- BIS, ECB, EUROSTAT, IMF, OECD, World Bank, UN (?), ILO, many NSOs...
- used in many different contexts (e.g NSO pushes to IO etc)
- lengthy 145 page User Guide Link
- only very small / relevant subset covered here (GET DATA)

How does the SDMX API work?

each call must have identifiers for:

- Statistical Agency (*Provider*)
- Database (*Dataflow*)
- Columns / Groupings (*Dimensions*), e.g.
  - → selected Countries
  - → Indicator you want
  - → Time Period
  - → **Dimensions differ for each dataset!**

A Basic call: chain all of these together to get the wanted dataset

very abstract, so lets see some

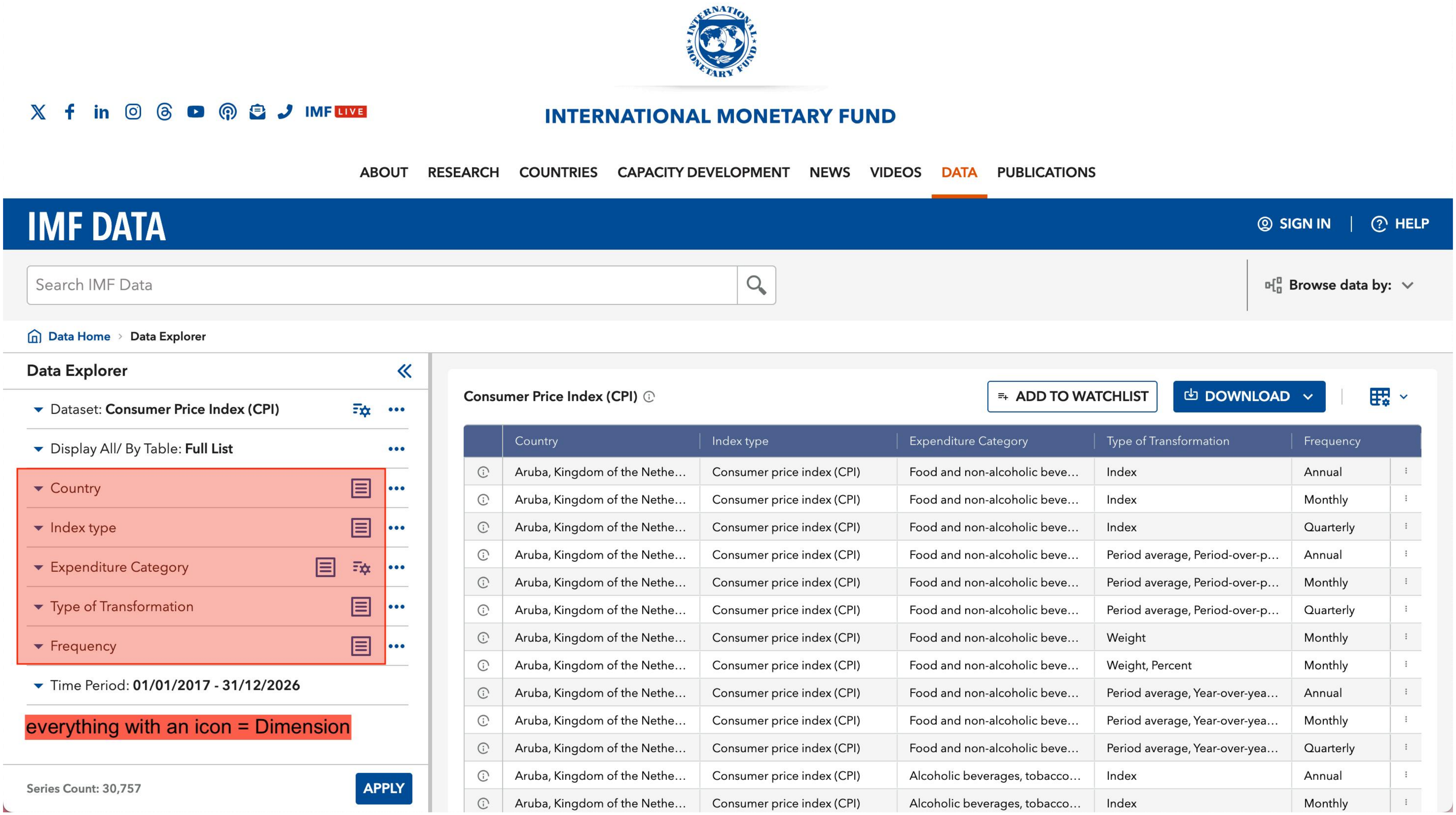
Examples

IMF

On the Website

Goal: CPI data since 1960 for USA, United Kingdom, and Germany

1. **data.imf.org**
2. **CPI Dataset**
3. **Data Explorer**



Data Explorer

- Dataset: **Consumer Price Index (CPI)**
- Display All/ By Table: **Full List**
- Country
- Index type
- Expenditure Category
- Type of Transformation
- Frequency
- Time Period: **01/01/2017 - 31/12/2026**

everything with an icon = Dimension

Series Count: 30,757

Consumer Price Index (CPI)

Country	Index type	Expenditure Category	Type of Transformation	Frequency
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Food and non-alcoholic beve…	Index	Annual
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Food and non-alcoholic beve…	Index	Monthly
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Food and non-alcoholic beve…	Index	Quarterly
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Food and non-alcoholic beve…	Period average, Period-over-p…	Annual
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Food and non-alcoholic beve…	Period average, Period-over-p…	Monthly
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Food and non-alcoholic beve…	Period average, Period-over-p…	Quarterly
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Food and non-alcoholic beve…	Weight	Monthly
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Food and non-alcoholic beve…	Weight, Percent	Monthly
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Food and non-alcoholic beve…	Period average, Year-over-yea…	Annual
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Food and non-alcoholic beve…	Period average, Year-over-yea…	Monthly
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Food and non-alcoholic beve…	Period average, Year-over-yea…	Quarterly
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Alcoholic beverages, tobacco…	Index	Annual
Aruba, Kingdom of the Nethe…	Consumer price index (CPI)	Alcoholic beverages, tobacco…	Index	Monthly

- In the Filter Tab, select categories for all dimensions (one or multiple)
- then click on Listing Icon and switch to Id
- take note of the IDs you want
- THE ORDER IS IMPORTANT!

Dimension	**Name**	**ID**
Country	United Stations, United Kingdom, Germany	USA, GBR, DEU
Index Type	Consumer Price Index	CPI
Expenditure	All Items	_T
Type of Transformation	Period average, Y-O-Y percentage change	YOY_PCH_PA_PT
Frequency	Annual	A

In the Code

First, lets import the relevant libraries:

```r
library(tidyverse)
library(rsdmx)
```

define our key by pasting together the selected IDs for each dimension

```r
COUNTRIES <- "USA+GBR+DEU" # All countries
INDICATOR <- "CPI" # selected Indicator
EXPENDITURE <- "_T" # All items / Total
TRANSFORMATION <- "YOY_PCH_PA_PT" # Percentage change
FREQUENCY <- "A" # Annual

key <- paste0(COUNTRIES, ".", INDICATOR,".", EXPENDITURE, ".", TRANSFORMATION, ".", FREQUENCY)

key
```

```
[1] "USA+GBR+DEU.CPI._T.YOY_PCH_PA_PT.A"
```

Now, we call the API, specifying our provider, database, key, and time period

```r
raw_data <- readSDMX(
  providerId = "IMF_DATA",  # IMF as Provider
  resource = "data",        # we want data
  flowRef = "CPI",          # from the CPI database
  key = key,                # our carefully created key
  start = 1960,             # lets limit it to start in 1960
  )
```

```
[rsdmx][INFO] Fetching
'https://api.imf.org/external/sdmx/2.1/data/CPI/USA+GBR+DEU.CPI._T.YOY_PCH_PA_PT.A/all/?startPeriod=1960'
```

The response is in SDMX format, so we have to convert it to a data frame first

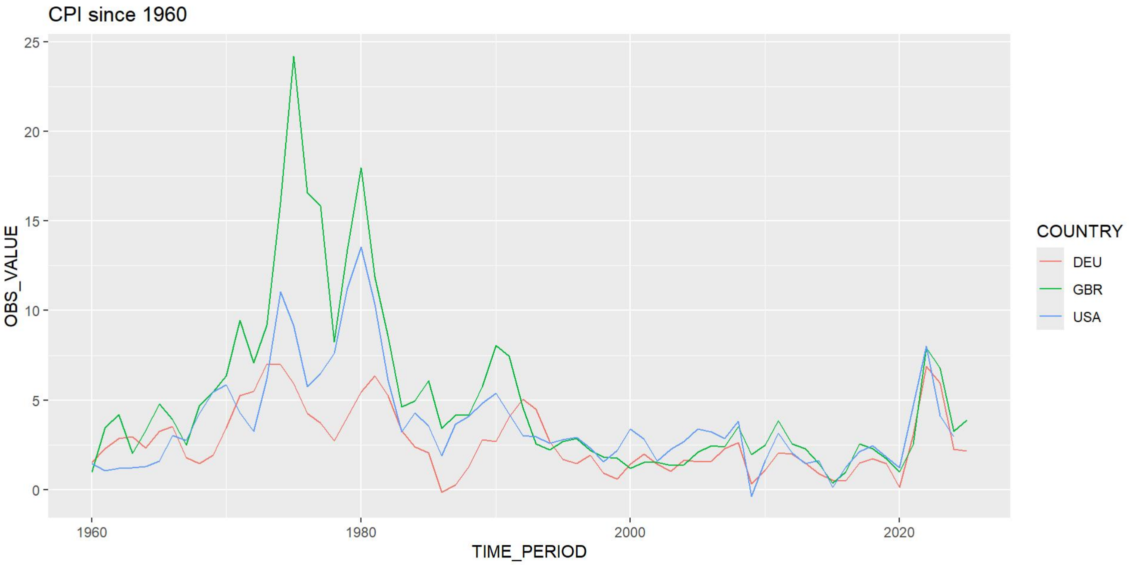
```r
df <- as.data.frame(raw_data)

head(df)
```

	COUNTRY <chr>	INDEX_TYPE <chr>	COICOP_1999 <chr>	TYPE_OF_TRANSFORMATION <chr>	FREQUENCY <chr>	IFS <cl
1	DEU	CPI	_T	YOY_PCH_PA_PT	A	tru
2	DEU	CPI	_T	YOY_PCH_PA_PT	A	tru
3	DEU	CPI	_T	YOY_PCH_PA_PT	A	tru
4	DEU	CPI	_T	YOY_PCH_PA_PT	A	tru
5	DEU	CPI	_T	YOY_PCH_PA_PT	A	tru
6	DEU	CPI	_T	YOY_PCH_PA_PT	A	tru

6 rows | 1-8 of 14 columns

Lets make a quick plot

```
df %>%
  select(COUNTRY, TIME_PERIOD, OBS_VALUE) %>%   # select relevant columns
  mutate(                                       # change types of columns
    OBS_VALUE = as.numeric(OBS_VALUE),
    TIME_PERIOD = as.numeric(TIME_PERIOD)
    ) %>%
  ggplot(aes(x=TIME_PERIOD, y=OBS_VALUE, color=COUNTRY)) +     # basic ggplot call
    geom_line() +                               # present as line plot
    labs(title="CPI since 1960")                # add title
```

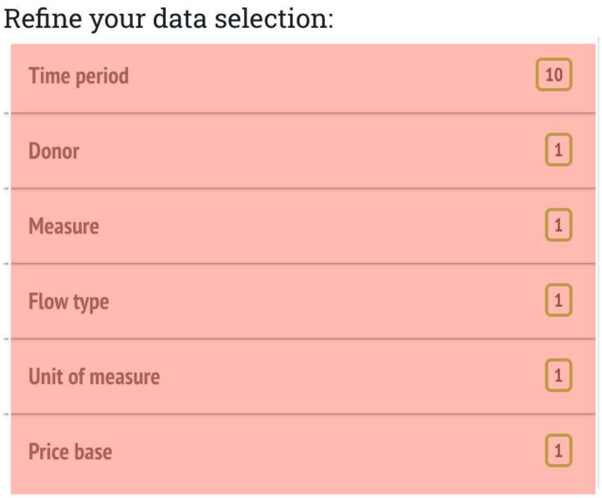


OECD

Goal: Official Development Assistance (ODA)

1. data-explorer.oecd.org/
2. DAC Table 1

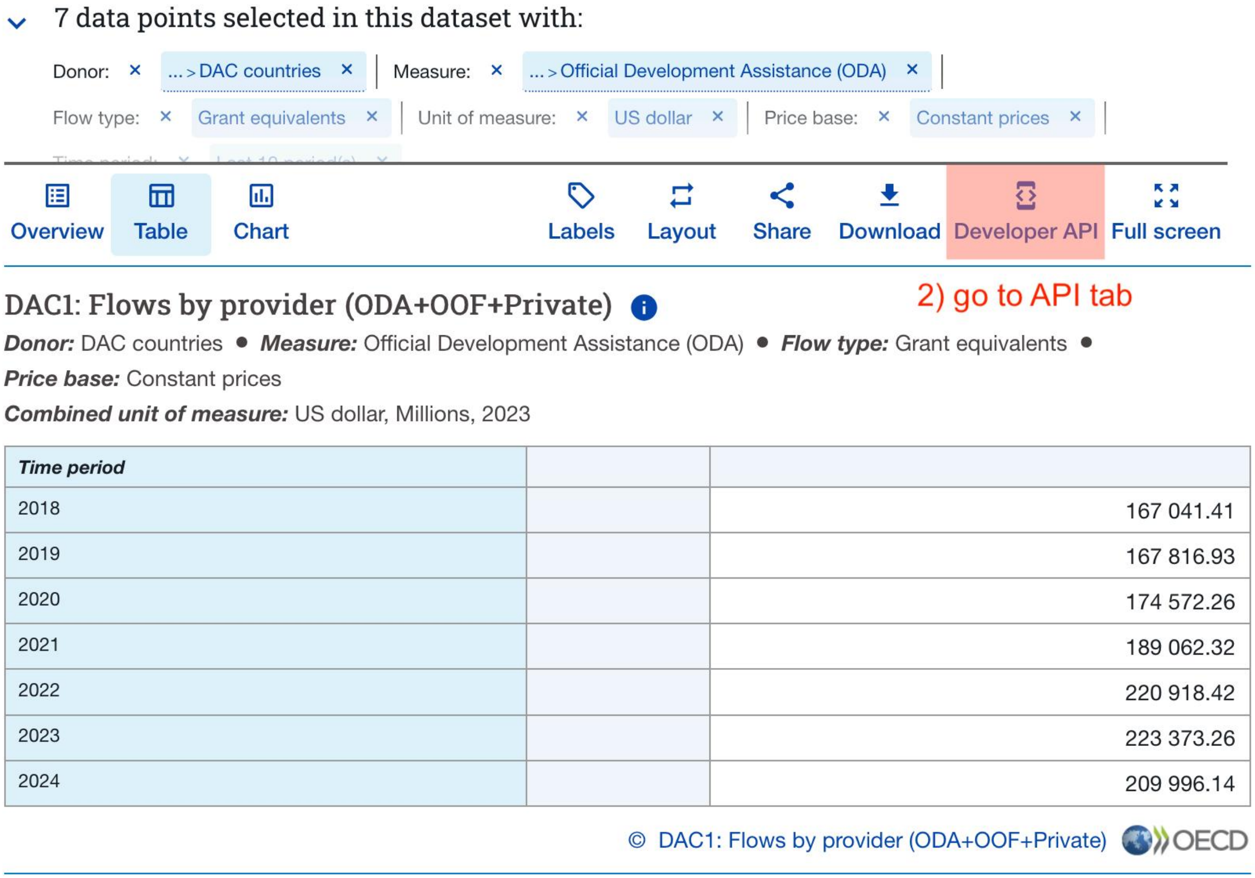
On the Website



Refine your data selection:

- Time period 10
- Donor 1
- Measure 1
- Flow type 1
- Unit of measure 1
- Price base 1

1) build a selection



7 data points selected in this dataset with:

Donor: × ... > DAC countries × | Measure: × ... > Official Development Assistance (ODA) ×

Flow type: × Grant equivalents × | Unit of measure: × US dollar × | Price base: × Constant prices ×

Time period: × Last 10 period(s) ×

2) go to API tab

DAC1: Flows by provider (ODA+OOF+Private)

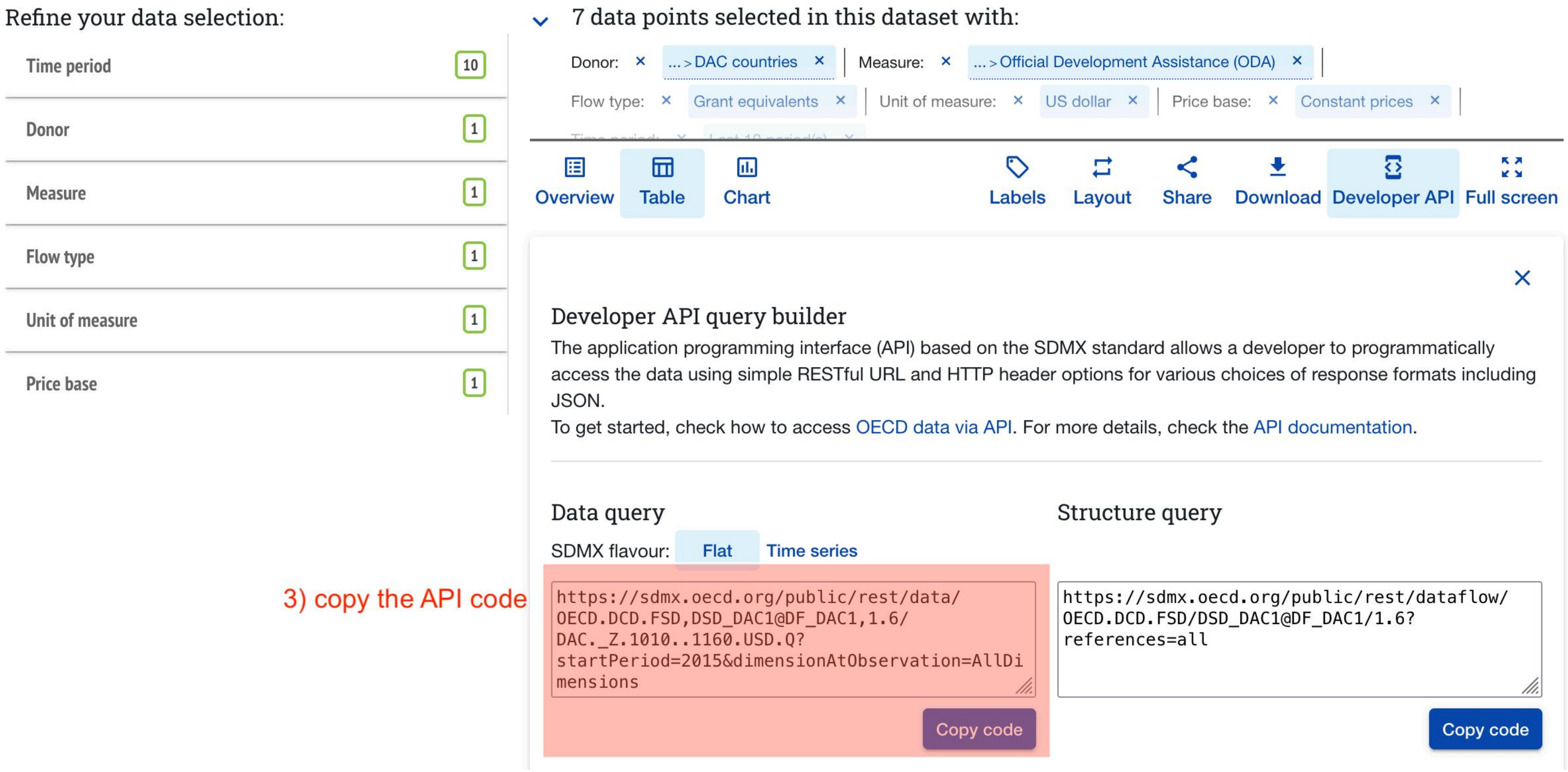
Donor: DAC countries ● ***Measure:*** Official Development Assistance (ODA) ● ***Flow type:*** Grant equivalents ●

Price base: Constant prices

Combined unit of measure: US dollar, Millions, 2023

Time period		
2018		167 041.41
2019		167 816.93
2020		174 572.26
2021		189 062.32
2022		220 918.42
2023		223 373.26
2024		209 996.14

© DAC1: Flows by provider (ODA+OOF+Private) OECD



Refine your data selection:

Time period	10
Donor	1
Measure	1
Flow type	1
Unit of measure	1
Price base	1

7 data points selected in this dataset with:

Donor: × … > DAC countries × | Measure: × … > Official Development Assistance (ODA) ×

Flow type: × Grant equivalents × | Unit of measure: × US dollar × | Price base: × Constant prices ×

Time period: × Last 10 period(s) ×

Developer API query builder

The application programming interface (API) based on the SDMX standard allows a developer to programmatically access the data using simple RESTful URL and HTTP header options for various choices of response formats including JSON.

To get started, check how to access OECD data via API. For more details, check the API documentation.

Data query

SDMX flavour: Flat Time series

```
https://sdmx.oecd.org/public/rest/data/
OECD.DCD.FSD,DSD_DAC1@DF_DAC1,1.6/
DAC._Z.1010..1160.USD.Q?
startPeriod=2015&dimensionAtObservation=AllDi
mensions
```

Structure query

```
https://sdmx.oecd.org/public/rest/dataflow/
OECD.DCD.FSD/DSD_DAC1@DF_DAC1/1.6?
references=all
```

3) copy the API code

In my Code

```r
url <- "https://sdmx.oecd.org/public/rest/data/OECD.DCD.FSD,DSD_DAC1@DF_DAC1,1.6/DAC._Z.1010..1160.USD.
```

now get the data and convert it to a data frame

```r
raw_data_oecd <- readSDMX(url)
df_oecd <- as.data.frame(raw_data_oecd)
```

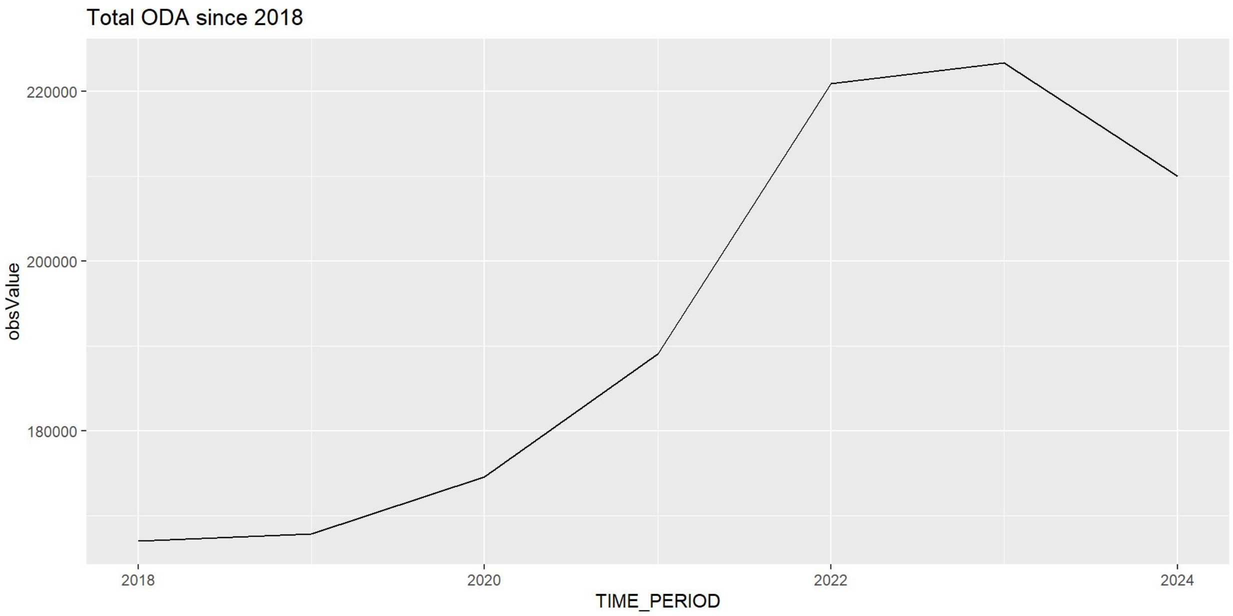
```r
head(df_oecd)
```

	TIME_PERIOD <chr>	DONOR <chr>	SECTOR <chr>	MEASURE <chr>	TYING_STATUS <chr>	FLOW_TYPE <chr>	UNIT_MEASURE <chr>
1	2018	DAC	_Z	1010	_Z	1160	USD
2	2023	DAC	_Z	1010	_Z	1160	USD
3	2022	DAC	_Z	1010	_Z	1160	USD
4	2024	DAC	_Z	1010	_Z	1160	USD
5	2019	DAC	_Z	1010	_Z	1160	USD
6	2020	DAC	_Z	1010	_Z	1160	USD

6 rows | 1-9 of 13 columns

a quick plot

```r
df_oecd %>%
  mutate(
    obsValue = as.numeric(obsValue),
    TIME_PERIOD = as.numeric(TIME_PERIOD)
  ) %>%
  ggplot(aes(x=TIME_PERIOD, y=obsValue)) +
  geom_line() +
  labs(title="Total ODA since 2018")
```



Exercise

Prerequisites

The RSDMX Package

```r
install.packages("rsdmx")
```

Your Task

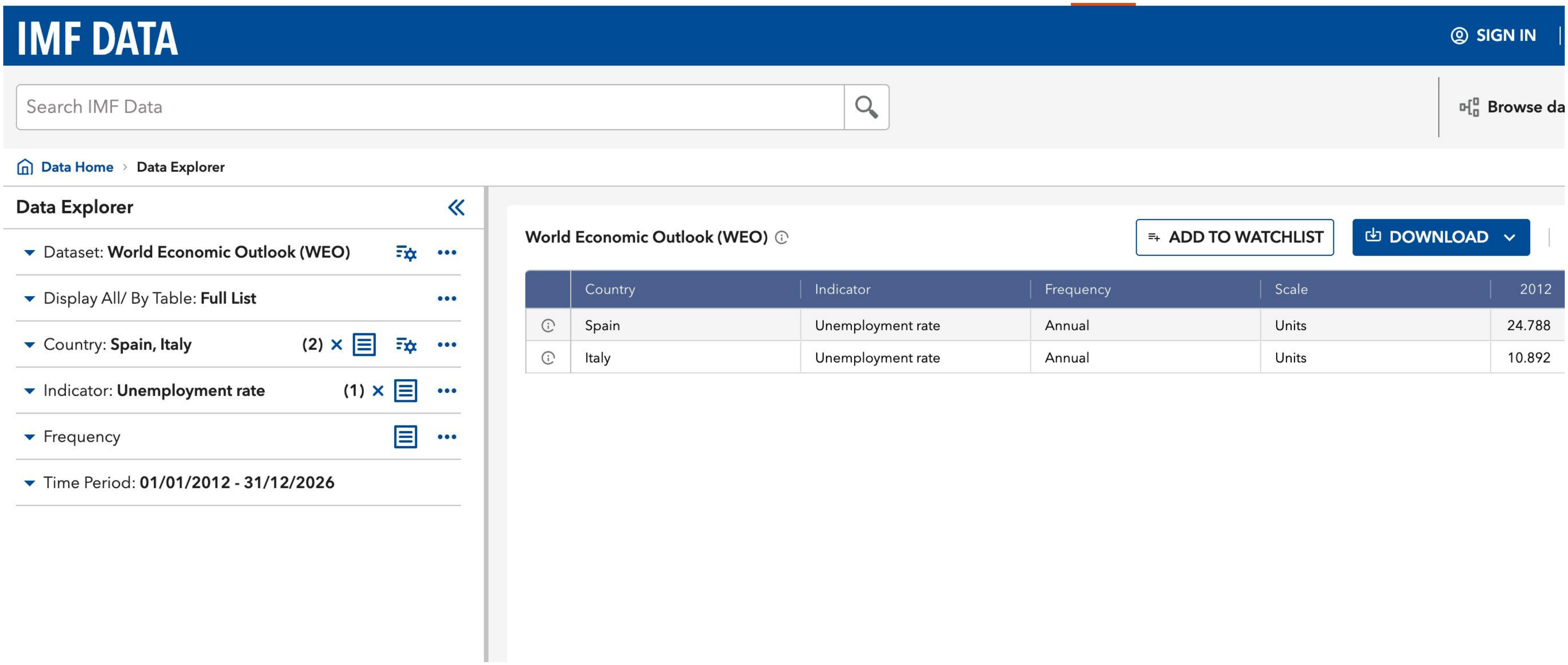
Unemployment Rate for Spain and Italy since 2012

- both ways possible
- different tradeoffs:
  - → IMF = much easier to search, but more difficult to access
  - → OECD = vice versa

Links:

- data.imf.org
- data-explorer.oecd.org

The Solution: IMF



Data Explorer

- Dataset: **World Economic Outlook (WEO)**
- Display All/ By Table: **Full List**
- Country: **Spain, Italy** (2)
- Indicator: **Unemployment rate** (1)
- Frequency
- Time Period: **01/01/2012 - 31/12/2026**

World Economic Outlook (WEO)

Country	Indicator	Frequency	Scale	2012
Spain	Unemployment rate	Annual	Units	24.788
Italy	Unemployment rate	Annual	Units	10.892

```r
COUNTRIES <- "ESP+ITA" # Spain and Italy
INDICATOR <- "LUR" # Unemployment Rate
FREQUENCY <- "A" # Annual

key <- paste0(COUNTRIES, ".", INDICATOR,".", FREQUENCY)
```

```r
raw_data_unemp <- readSDMX(
  providerId = "IMF_DATA",
  resource = "data",
  flowRef = "WEO", # World Economic Outlook
  key = key,
  start = 2012,
  )
```

```
[rsdmx][INFO] Fetching 'https://api.imf.org/external/sdmx/2.1/data/WEO/ESP+ITA.LUR.A/all/?startPeriod=2012'
```

```r
df_unemp <- as.data.frame(raw_data_unemp)
df_unemp %>%
  select(COUNTRY, TIME_PERIOD, OBS_VALUE) %>%
  pivot_wider(names_from=COUNTRY, values_from=OBS_VALUE)
```

TIME_PERIOD	ESP	ITA
<chr>	<chr>	<chr>
2012	24.788	10.892
2013	26.095	12.367
2014	24.443	12.783
2015	22.058	12.017
2016	19.635	11.717
2017	17.225	11.3
2018	15.255	10.6
2019	14.105	9.908
2020	15.533	9.367
2021	14.92	9.525

1-10 of 19 rows

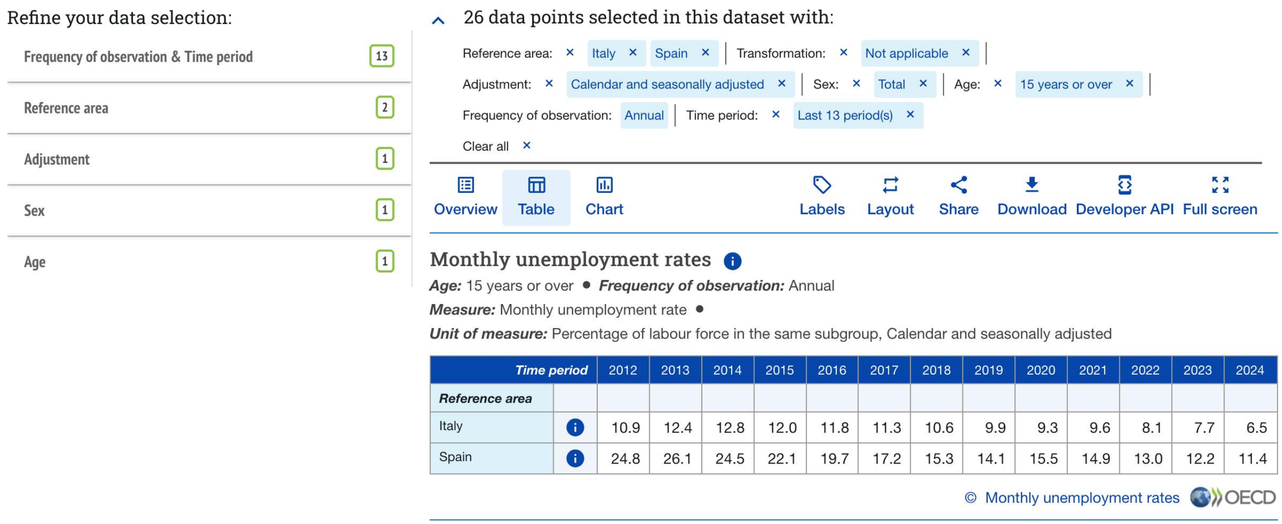
Previous 1 2 Next

The Solution: Eurostat

Search for “Unemployment Rate” on data-explorer.oecd.org

Misleading: namend “monthly Unemployment Rate”, but has annual frequency as well

This Table has everything we need: Link



26 data points selected in this dataset with:

Reference area: × Italy × Spain × | Transformation: × Not applicable × |
Adjustment: × Calendar and seasonally adjusted × | Sex: × Total × | Age: × 15 years or over × |
Frequency of observation: Annual | Time period: × Last 13 period(s) ×
Clear all ×

Monthly unemployment rates

Age: 15 years or over ● ***Frequency of observation:*** Annual
Measure: Monthly unemployment rate ●
Unit of measure: Percentage of labour force in the same subgroup, Calendar and seasonally adjusted

Time period		2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024
Reference area														
Italy	i	10.9	12.4	12.8	12.0	11.8	11.3	10.6	9.9	9.3	9.6	8.1	7.7	6.5
Spain	i	24.8	26.1	24.5	22.1	19.7	17.2	15.3	14.1	15.5	14.9	13.0	12.2	11.4

© Monthly unemployment rates OECD

```r
url <- "https://sdmx.oecd.org/public/rest/data/OECD.SDD.TPS,DSD_LFS@DF_IALFS_UNE_M,1.0/ESP+ITA..._Z.Y._
```

```r
raw_data_eurostat <- readSDMX(url)
df_eurostat <- as.data.frame(raw_data_eurostat)
df_eurostat %>%
  select(REF_AREA, TIME_PERIOD, obsValue) %>%
  pivot_wider(names_from=REF_AREA, values_from=obsValue)
```

TIME_PERIOD	ITA	ESP
<chr>	<dbl>	<dbl>
2012	10.883330	24.79167
2013	12.366670	26.11667
2014	12.825000	24.45000
2015	12.000000	22.07500
2016	11.750000	19.65000
2017	11.258330	17.23333
2018	10.600000	15.26667
2019	9.933333	14.10833
2020	9.300000	15.53333
2021	9.558333	14.93333

1-10 of 13 rows

Previous 1 2 Next

Helpful Tips

Github Copilot

Coding Assistant

- with **free tier**
- directly integrated in RStudio, completes your code on TAB

Note:

- not well trained on (SDMX) API calls, so you probably have to build it yourself
- tidyverse = works really good (although sometimes quite inefficient)

Quarto Markdown

New format in RStudio

- combines Prose and Code
- very good for documenting code and pasting links etc
- allows to produce PDFs, HTML, slides, …

Tidyverse Cheat Sheets

All packages in the tidyverse in R have very good cheat sheets!

- **dplyr** (for transforming data): PDF
- **ggplot** (data visualisation): PDF
- **tidyr** (for cleaning data): PDF

Links

- A gentle introduction to SDMX for reproducible data extraction from international organizations
- imfapi package for R (quite new, but should work)
- IMF SDMX central

  → verz convoluted overview of IMF datasets, but able to see dimensions
- Eurostat Query Builder
- ILO query builder

additional code

list of all providers

```r
providers <- getSDMXServiceProviders()
as.data.frame(providers)
```

agencyId <chr>	name <chr>
BIS	Bank for International Settlements
ECB	European Central Bank
ESTAT	Eurostat (Statistical office of the European Union)
ESTAT	Eurostat (Statistical office of the European Union) - COMEXT
ESTAT	Eurostat (Statistical office of the European Union) - DG COMP
ESTAT	Eurostat (Statistical office of the European Union) - DG GROW
ESTAT	Eurostat (Statistical office of the European Union) - DG EMPL
IMF	International Monetary Fund
IMF_DATA	International Monetary Fund - Data Portal
OECD	Organisation for Economic Cooperation and Development

1-10 of 29 rows | 1-2 of 6 columns

Previous 1 2 3 Next

add labels to data frame

```r
COUNTRIES <- "*" # All countries
INDICATOR <- "CPI" # selected Indicator
EXPENDITURE <- "_T" # All items / Total
TRANSFORMATION <- "YOY_PCH_PA_PT" # Percentage change
FREQUENCY <- "A" # Annual
key <- paste0(COUNTRIES, ".", INDICATOR,".", EXPENDITURE, ".", TRANSFORMATION, ".", FREQUENCY)
```

```r
raw_data <- readSDMX(
  providerId = "IMF_DATA",  # IMF as Provider
  resource = "data",        # we want data
  flowRef = "CPI",          # from the CPI database
  key = key,                # our carefully created key
  start = 1960,             # lets limit it to start in 1960
  dsd = TRUE
  )
```

```
[rsdmx][INFO] Fetching 'https://api.imf.org/external/sdmx/2.1/data/CPI/*.CPI._T.YOY_PCH_PA_PT.A/all/?
startPeriod=1960'
[rsdmx][INFO] Attempt to fetch DSD ref from dataflow description
[rsdmx][INFO] Fetching 'https://api.imf.org/external/sdmx/2.1/dataflow/all/CPI/latest/'
[rsdmx][INFO] Fetching 'https://api.imf.org/external/sdmx/2.1/datastructure/all/DSD_CPI/latest/?
references=descendants'
[rsdmx][INFO] DSD fetched and associated to dataset!
```

```r
df <- as.data.frame(raw_data, label=T)
head(df)
```

	COUNTRY <chr>	COUNTRY_label.fr <chr>	COUNTRY_label.ar <chr>
1	ABW	Aruba, Royaume des Pays-Bas	أروبا، مملكة هولندا
2	ABW	Aruba, Royaume des Pays-Bas	أروبا، مملكة هولندا
3	ABW	Aruba, Royaume des Pays-Bas	أروبا، مملكة هولندا
4	ABW	Aruba, Royaume des Pays-Bas	أروبا، مملكة هولندا
5	ABW	Aruba, Royaume des Pays-Bas	أروبا، مملكة هولندا
6	ABW	Aruba, Royaume des Pays-Bas	أروبا، مملكة هولندا

6 rows | 1-4 of 45 columns

download complete tables

you can also download complete tables without specifying keys (Beware, it takes a lot of time!)

```r
imf_raw <- readSDMX(
  providerId = "IMF_DATA",
  resource = "data",
  flowRef = "WEO",
  start = 2023,
  end = 2025
  )
```

```
[rsdmx][INFO] Fetching 'https://api.imf.org/external/sdmx/2.1/data/WEO/all/all/?
startPeriod=2023&endPeriod=2025'
```

```r
df_huge <- as.data.frame(imf_raw)
head(df_huge)
```

	COUNTRY	INDICATOR	FREQUENCY	COUNTRY_UPDATE_DATE	OVERLAP	DECIMALS_DISPL
	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>
1	ABW	BCA	A	9/19/2025	OL	3
2	ABW	BCA	A	9/19/2025	OL	3
3	ABW	BCA	A	9/19/2025	OL	3
4	ABW	BCA_NGDPD	A	9/19/2025	OL	3
5	ABW	BCA_NGDPD	A	9/19/2025	OL	3
6	ABW	BCA_NGDPD	A	9/19/2025	OL	3

6 rows | 1-8 of 10 columns