United Nations Data Portal
Population Division

Data API

▶ IntroductionIndicatorsTopicsLocationsDataData SourcesTutorial using PythonTutorial using R

DataPortal API

Introduction

Users can also access data on the Data Portal via its API for displaying on webpag

Base Path

The base path for accessing the API is:

`https://population.un.org/dataportalapi/api/v1`

All calls to the API will use this base path, plus a relative path that may include the

The default response format for the API is JSON, but the API also supports other o

Status codes

Status codes are useful when calling the API, as they can inform the user of whethe

```
200 : Successful request

400 : Bad request

404 : Input parameters not found

406 : Requested output format not allowed

500 : Server error
```

Structure of API response

When a user calls the API, the JSON formatted response will return the following in

`pageNumber` : the current page of the response, which may have multiple pages

`pageSize` : the number of records returned on the current page (a maximum of 10

`previousPage` : the path to the previous page of the response when multiple pag

`nextPage` : the path to the next page of the response when multiple pages are retu

`pages` : the total number of pages in the response

`total` : the total number of records in the response

`data` : the actual data returned in the response

Indicators

Users can get information on the indicators available on the Data Portal using the r

`/indicators/{codes}`

The `/{codes}` suffix is an option that can be used to restrict the reponse to only ir

The retrieved information include: the indicator ID number, full name, description of

Users can access a complete list of indicators available through the API by making

`https://population.un.org/dataportalapi/api/v1/indicators`

Users can also limit their call to only a specific indicator or a list of indicators. They `https://population.un.org/dataportalapi/api/v1/indicators/CPA`

To make the same call using the indicator ID number, one would specify:

`https://population.un.org/dataportalapi/api/v1/indicators/1?s`

The response to both of these calls in JSON format would be:

```
[{'id': 1,
  'name': 'Contraceptive prevalence: Any method (Percen
  'shortName': 'CPAnyP',
  'description': 'Percentage of women of reproductive a
  'displayName': 'Any',
  'dimAge': False,
  'dimSex': False,
  'dimVariant': True,
  'dimCategory': True,
  'defaultAgeId': 31,
  'defaultSexId': 2,
  'defaultVariantId': 4,
  'defaultCategoryId': 100,
  'variableType': 'relative',
  'valueType': 'percent',
  'unitScaling': 0.01,
  'precision': 1,
  'isThousandSeparatorSpace': False,
  'formatString': '#0.0',
  'unitShortLabel': '%',
  'unitLongLabel': 'per cent',
  'nClassesDefault': 5,
  'downloadFileName': 'PercentageContraceptive_AnyMetho
  'sourceId': 23,
  'sourceName': 'Estimates and Projections of Family Pl
  'sourceYear': 2022,
  'sourceStartYear': 1970,
  'sourceEndYear': 2030,
  'sourceCitation': 'United Nations, Department of Econ
  'sourceUrl': 'https://www.un.org/development/desa/pd/
  'topicId': 5,
  'topicName': 'Family Planning',
  'topicShortName': 'FP'}]
```

Below is an example in which information on a list of indicators are retrieved. Using

`https://population.un.org/dataportalapi/api/v1/indicators/MAC`

And using indicator ID values:

`https://population.un.org/dataportalapi/api/v1/indicators/18,`

Topics

All indicators are categorized according to specific topics, and these topics may th

`/topics/{topicIDorShortName}/indicators/{IndicatorIDorShortNa`

To retrieve a complete list of topics covered in the Data Portal, one could call:

`https://population.un.org/dataportalapi/api/v1/topics?sort=so`

which would return:

```

[{'id': 0, 'name': 'Not applicable', 'shortName': 'NA',
 {'id': 1, 'name': 'Population', 'shortName': 'Pop', 's
 {'id': 2, 'name': 'Fertility', 'shortName': 'Fert', 's
 {'id': 3, 'name': 'Mortality', 'shortName': 'Mort', 's
 {'id': 4, 'name': 'International Migration', 'shortNam
 {'id': 5, 'name': 'Family Planning', 'shortName': 'FP'
 {'id': 6, 'name': 'Marital Status', 'shortName': 'MarS
 {'id': 7, 'name': 'All Components', 'shortName': 'All'
 {'id': 8, 'name': 'Child Mortality', 'shortName': 'IGM
 {'id': 9, 'name': 'Maternal Mortality', 'shortName': '
```

Using a short name or ID from this list of topics, one could then find all relevant ind

Topic short name: `https://population.un.org/dataportalapi/api/v1/`

ID number: `https://population.un.org/dataportalapi/api/v1/topics`

Below is a call to retrieve all indicators available under the topics Population (shor

`https://population.un.org/dataportalapi/api/v1/topics/Pop,Mar`

Locations

To retrieve a list of geographical areas included in the Data Portal, they can use the

`/locations/{codes}`

As previously, the `/{codes}` suffix is optional if one would like only geographic inf

The returned information includes: location ID, parent region ID, full name, ISO2 and

To retrieve a full list of geographical areas covered in the Data Portal, the user woul

`https://population.un.org/dataportalapi/api/v1/locations?sort`

A user may also retrieve information only for one or a list of geographical areas usi

ID number: `https://population.un.org/dataportalapi/api/v1/locat`

ISO2: `https://population.un.org/dataportalapi/api/v1/locations/`

ISO3: `https://population.un.org/dataportalapi/api/v1/locations/`

All three of these calls will return the following JSON output:

```
[{'id': 704,
  'parentId': 920,
  'name': 'Viet Nam',
  'iso3': 'VNM',
  'iso2': 'VN',
  'locationTypeId': 4,
  'locationType': 'Country',
  'longitude': 108.27719879150392,
  'latitude': 14.058323860168455}]
```

Below is an example in which a list of geographical areas is supplied for Ghana, Inc

ID number: `https://population.un.org/dataportalapi/api/v1/locat`

ISO2: `https://population.un.org/dataportalapi/api/v1/locations/`

Aggregates

To obtain information on the various aggregates (such as SDG Regions, World Banl

`/locationsWithAggregates`

The full query would thus be:

`https://population.un.org/dataportalapi/api/v1/locationsWithA`

The response to this call will provide the same basic information included in the `/l`

Data

Most users will want to be able to call the API to return a specific set of indicators f

`/data/indicators/{indicators}/locations/{locations}/start/{st`

Using the information already covered on accessing location codes and indicator c

The example below retrieves data on the percentage of women who are currently m

`https://population.un.org/dataportalapi/api/v1/data/indicator`

To retrieve data on an additional two countries, Saudi Arabia (location ID: 682) and

`https://population.un.org/dataportalapi/api/v1/data/indicator`

The query could also be extended to return more than one indicator. In this exampl

`https://population.un.org/dataportalapi/api/v1/data/indicator`

To obtain information on the various aggregates (such as SDG Regions, World Bank

`/locationsWithAggregates`

The full query would thus be:

`https://population.un.org/dataportalapi/api/v1/locationsWithA`

If instead a user would like to have the structure of the response be one in which ro

`/locationsWithFlatAggregates`

The full query would be:

`https://population.un.org/dataportalapi/api/v1/locationsWithF`

Both of these paths can be ammended with specific geographical codes if a user w

`https://population.un.org/dataportalapi/api/v1/locationsWithA`

`https://population.un.org/dataportalapi/api/v1/locationsWithF`

Query String Parameters

In addition to the query, users may specify parameters indicating how the API respo

`sort` allows users to order the data with respect to a retrieved column. For exampl

`https://population.un.org/dataportalapi/api/v1/indicators?sor`

If one would like these to be sorted in reverse alphabetical order, the query would in

`https://population.un.org/dataportalapi/api/v1/indicators` http

`format` allows users to specify the format of the data retrieved from the data porta

`https://population.un.org/dataportalapi/api/v1/indicators?sor`

Note that records returned in a CSV format will use the `|` separator.

Pagination

Sometimes a query using the `/data` endpoint will return a large number of results.

`https://population.un.org/dataportalapi/api/v1/data/indicator`

Because of the default settings, this query would only return 100 results. If instead

`https://population.un.org/dataportalapi/api/v1/data/indicator`

When calling the `/data` endpoint, the body of the response will include the followir

`pageNumber` - indicates the current page of the response. By deault this will be pa

`pageSize` - indicates the total number of pages in the response. The default value

`nextPage` - the path for the next page of the query

`previousPage` - the path for the previous page of the query

`total` - the total number of datapoints returned by the call

`pages` - the total number of pages returned by the call

`data` - This is the part of the response that includes the data returned from the que

If a user prefers that paging information is not included in the response body, but ir

```
    'X-Pagination': '{"PageNumber":1,"PageSize":100,"Ne
```

In the following example, we can see that the results that have been returned conta

`https://population.un.org/dataportalapi/api/v1/data/indicator`

The above query will then return the next 100 results of the query, and pageNumbe

Data Sources

Users may also retrieve information on the sources of the various indicators availal

`https://population.un.org/dataportalapi/api/v1/sources?sort=i`

The API can also be called to return all indicators associated with a given source by

For example, to find all indicators associated with the the *2022 Revision of World F*

`https://population.un.org/dataportalapi/api/v1/sources/25/ind`

Tutorial using Python

Basic Examples

In this section, some examples will be provided to show how to access the Data Po

Tutorial using Python

Basic Examples

In this section, some examples will be provided to show how to access the Data Po

Example 1: Returning a list of indicators

In the first example, a call will be made to the API to return a complete list of indica

```python
import pandas as pd
import requests
import json

# Declares the base url for calling the API
base_url = "https://population.un.org/dataportalapi/api

# Creates the target URL, indicators, in this instance
target = base_url + "/indicators/"

# Get the response, which includes the first page of da
```

```
response = requests.get(target)

# Converts call into JSON
j = response.json()

# Converts JSON into a pandas DataFrame.
df = pd.json_normalize(j['data']) # pd.json_normalize f

# Loop until there are new pages with data
while j['nextPage'] != None:
    # Reset the target to the next page
    target = j['nextPage']

    #call the API for the next page
    response = requests.get(target)

    # Convert response to JSON format
    j = response.json()

    # Store the next page in a data frame
    df_temp = pd.json_normalize(j['data'])

    # Append next page to the data frame
    df = df.append(df_temp)
```

To view response code:

```
    print(response)

    ##  <Response [200]>
```

Example 2: Returning a list of geographical areas

In this example, the list of geographical areas are retrieved from the API:

```
# Creates the target URL, indicators, in this instance
target = base_url + "/locations/"

# Get the response, which includes the first page of da

response = requests.get(target)

# Converts call into JSON
j = response.json()

# Converts JSON into a pandas DataFrame.
df = pd.json_normalize(j['data']) # pd.json_normalize f

# Loop until there are new pages with data
while j['nextPage'] != None:
    # Reset the target to the next page
    target = j['nextPage']

    #call the API for the next page
    response = requests.get(target)

    # Convert response to JSON format
    j = response.json()

    # Store the next page in a data frame
    df_temp = pd.json_normalize(j['data'])

    # Append next page to the data frame
    df = df.append(df_temp)

```

Example 3: Returning a single indicator for a single geographical area

The following example calls the API to return the contraceptive prevalence rate (inc

```
# Creates the target URL, indicators, in this instance
target = base_url + "/data/indicators/1/locations/4/sta

# Get the response, which includes the first page of da
response = requests.get(target)

# Converts call into JSON
j = response.json()

# Converts JSON into a pandas DataFrame.
df = pd.json_normalize(j['data']) # pd.json_normalize f

# Loop until there are new pages with data
while j['nextPage'] != None:
    # Reset the target to the next page
    target = j['nextPage']

    #call the API for the next page
    response = requests.get(target)

    # Convert response to JSON format
    j = response.json()

    # Store the next page in a data frame
    df_temp = pd.json_normalize(j['data'])

    # Append next page to the data frame
    df = df.append(df_temp)
```

And to view only data for all women and the median estimates:

```
df2 = df[(df['variant']=="Median") & (df["category"]=='
```

Example 4: Returning data on multiple indicators and geographical areas

Below is one more example using a more complicated search in which a user wish

```
    # Define a function that will take a relative path
    def callAPI(relative_path:str, topic_list:bool = Fa
        base_url = "https://population.un.org/dataporta
        target = base_url + relative_path # Query strin
        # Calls the API
        response = requests.get(target)
        # Reformats response into a JSON object
        j = response.json()
        # The block below will deal with paginated resu
        # If results not paginated, this will be skippe
        try:
          # If results are paginated, they are transfor
          # The data may be accessed using the 'data' k
            df = pd.json_normalize(j['data'])
            # As long as the nextPage key of the dictio
            while j['nextPage'] is not None:
                response = requests.get(j['nextPage'])
                j = response.json()
                df_temp = pd.json_normalize(j['data'])
                df = df.append(df_temp)
        except:
            if topic_list:
                df = pd.json_normalize(j, 'indicators'
            else:
                df = pd.DataFrame(j)
        return(df)

    # Uses callAPI function to get a list of locations
    df_locations = callAPI("/locations/")

    # Identifies ID code for Western Africa
    western_africa_id = df_locations.loc[df_locations['

    # Restricts the dataframe to only include geographi
    df_locations = df_locations[df_locations['parentId'

    # Stores country codes in a list
    country_codes = [str(code) for code in df_locations

    # Converts country code list into a string to be us
    country_selection_string = ",".join(country_codes)

    # Uses callAPI function to get a list of Family Pla
    df_topics = callAPI("/topics/FP/indicators", topic_

    # Stores indicator codes in a list
    indicator_codes = [str(code) for code in df_topics[

    # Converts indicator code list into string to be us
    indicator_selection_string = ",".join(indicator_co

    # Calls the API to return the indicator values for
    df = callAPI(f"/data/indicators/{indicator_selecti

    # Finally, filters the returned results to only in
    df2 = df.loc[(df['variant']=="Median") & (df['categ
```

Detailed Examples

This section provides some examples in Python of how to access demographic in

Input

As a first step, it is necessary to import all the Python packages needed to run the

```
import pandas as pd
import json
import requests
import matplotlib.pyplot as plt
```

UNPD Data Portal

The data included in the Data Portal can be accessed using the following base URL

```
base_url_UNPD = "https://population.un.org/dataportala
```

The topics covered by the Data Portal can be found in the following way:

```
target = base_url_UNPD + "/topics/" # Define target URl

response = requests.get(target) # Call the API

j = response.json() # Convert response into JSON objec1

df = pd.json_normalize(j['data']) # convert JSON to dat
```

id	
0	Not applicabl
1	Population
2	Fertility
3	Mortality
4	International
5	Family Planni
6	Marital Statu:
7	All Componei
8	Child Mortalit
9	Maternal Moı

To find the geographical areas covered:

```
target = base_url_UNPD + "/locations/" # Define target

response = requests.get(target) # Call the API

j = response.json() # Convert response into JSON objec1

df = pd.json_normalize(j['data']) # convert JSON to dat
```

id	parentId
4	5501
8	925
12	912
16	957
20	925

For each geographical area, there are different groupings and aggregates (SDG Reg

```
target = base_url_UNPD + "/locationsWithAggregates/" #

response = requests.get(target) # Call the API

j = response.json() # Convert response into JSON objec1

df = pd.json_normalize(j['data']) # convert JSON to dat
```

FIELD1	id	
1	4	Afghanistan
2	4	Afghanistan
3	4	Afghanistan
4	4	Afghanistan
5	4	Afghanistan
6	8	Albania

World Bank Open Data

The first examples involve the use of the data queried through the World Bank Ope

```
base_url_WB = "http://api.worldbank.org/v2/"
```

The datasets are organized by topics:

```
target = base_url_WB + "/topic?format=json" # Define Wo

response = requests.get(target)
j = response.json()
df = pd.json_normalize(j[1])
```

id	value	
1	Agriculture & Rural Development	For the 70 percent of the world's poor who l
2	Aid Effectiveness	Aid effectiveness is the impact that aid has
3	Economy & Growth	Economic growth is central to economic de
4	Education	Education is one of the most powerful instr
5	Energy & Mining	The world economy needs ever-increasing a

Within each topic there are multiple indicators available. The following code loads t

```
target = base_url_WB + "/topic/11/indicator?format=jsor

response = requests.get(target)
j = response.json()
df = pd.json_normalize(j[1])
```

Here is a look at an extract of the possible indicators included:

id	value	
1	Agriculture & Rural Development	For the 70 percent of the world's poor who l
2	Aid Effectiveness	Aid effectiveness is the impact that aid has
3	Economy & Growth	Economic growth is central to economic de
4	Education	Education is one of the most powerful instr

id	value	
5	Energy & Mining	The world economy needs ever-increasing a

Example 1. Adolescent fertility vs. net enrollment in primary school, Senegal

This example illustrates an exploratory comparison between the trends of the adol

```

country_iso3 = "SEN" # Set the desired country
indicator_code_WB = "SE.PRM.NENR" # Set the desired inc

target = base_url_WB + f"/country/{country_iso3}/indica

response = requests.get(target) # Call WB API
j = response.json() # Create JSON object
pages = j[0]['pages'] # Identify number of pages in res

#Convert first page into a DataFrame
df_WB = pd.json_normalize(j[1])

#Loop through pages and append results to DataFrame
for page in range(2,pages+1):
    target = base_url_WB + f"/country/{country_iso3}/ir
    response = requests.get(target)
    j = response.json()
    df_temp = pd.json_normalize(j[1])
    df_WB = df_WB.append(df_temp)

# Verify that the length of the DataFrame is equal to t
assert len(df_WB)==j[0]['total'], "DataFrame observatic
```

countryiso3code
SEN
SEN
SEN
SEN
SEN

The data on fertility rates for women aged 15-19, instead, can be accessed through

```
country_M49 = 686 #set the country code
indicator_code_UNPD = 17 #set the indicator code
start_year =  1970 #set the start year
end_year =  2020 #set the end year

#define the target URL
target = base_url_UNPD + f"/data/indicators/{indicator_

response = requests.get(target) #Call the API
j = response.json() #Format response as JSON
df_UNPD = pd.json_normalize(j['data']) #Read JSON data

# As long as the response contains information in the '
while j['nextPage'] is not None:
    response = requests.get(j['nextPage'])
    j = response.json()
    df_temp = pd.json_normalize(j['data'])
    df_UNPD = df_UNPD.append(df_temp)

# Verifies that the number of records available from AP
assert len(df_UNPD)==j['total'], "DataFrame observatio

#Filter data to only include women between ages 15 and
df_UNPD = df_UNPD.loc[(df_UNPD['variant']=="Median") &
```

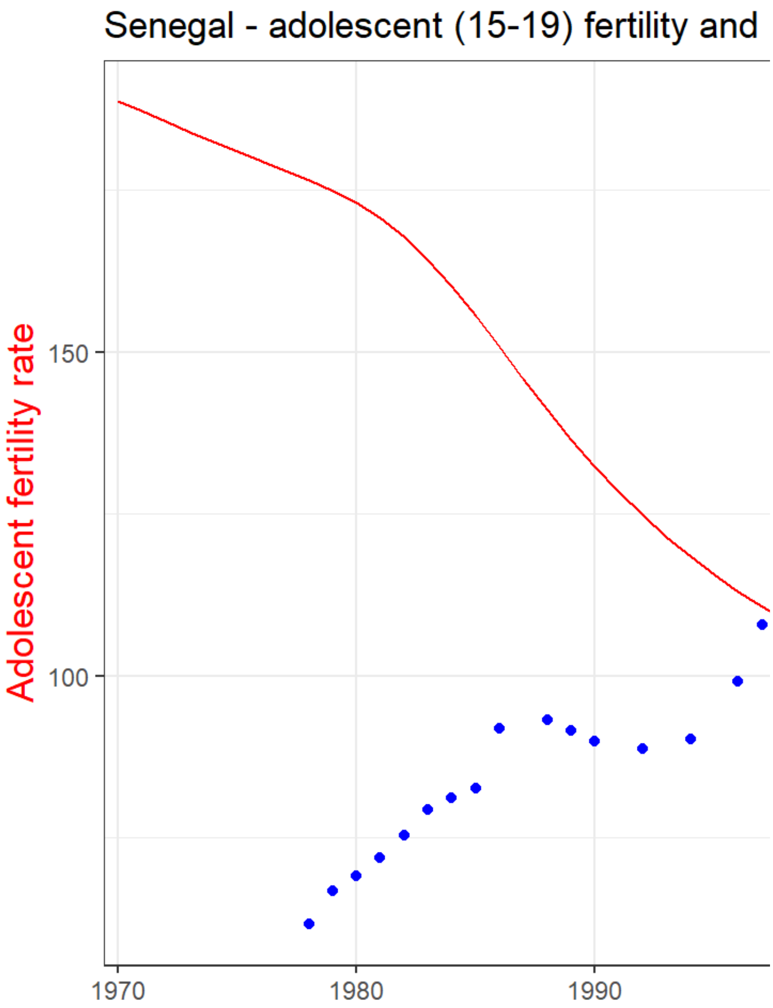
locationId	location	iso3	iso2	locationTypeId	
686	Senegal	SEN	SN	4	1
686	Senegal	SEN	SN	4	1
686	Senegal	SEN	SN	4	1
686	Senegal	SEN	SN	4	1
686	Senegal	SEN	SN	4	1

The two datasets obtained through the APIs then can be merged together. The mat

```
df_UNPD = df_UNPD.rename(columns={"timeLabel":"year", '
df_WB = df_WB.rename(columns={"countryiso3code":"iso3",

# Merge dataframes
df = pd.merge(df_UNPD[["location","iso3","year","ASFR"]

# convert year to int
df['year'] = df['year'].astype(int)
```

Now it is possible to plot the two indicators together to explore their trends over tin

```
# Creates plot
fig, ax = plt.subplots() # Instantiate figure and axes
fig.suptitle("Senegal\nAdolescent (15-19) fertility anc

ax.plot(df['year'], df['ASFR'], c='r') # create line pl
ax.set_ylabel("Adolescent fertility rate", color="r") #
ax.xaxis.set_ticks([i for i in range(1970, 2030, 10)])

ax2 = ax.twinx() # duplicate plot 1 x axis
ax2.scatter(df['year'], df['Enrollment'], color='b') #
ax2.set_ylabel("Net enrollment in primary school", colo
```



Example 2. Modern contraceptive prevalence and GDP per capita, Kenya

This example uses the data on modern contraceptive prevalence and on GDP per c

```

country_iso3 = "KEN" #set the ISO 3 code for the desire
indicator_code_WB = "NY.GDP.PCAP.CD" #set the code for

target = base_url_WB + f"/country/{country_iso3}/indica

response = requests.get(target) # Call the World Bank A
j = response.json()


pages = j[0]['pages'] # Identify total number of pages
df_WB = pd.json_normalize(j[1])

# loop through the pages in the response and append to
for page in range(2,pages+1):
    target = base_url_WB + f"/country/{country_iso3}/ir
    response = requests.get(target)
    j = response.json()
    df_temp = pd.json_normalize(j[1])
    df_WB = df_WB.append(df_temp)

#verify that the number of rows in the dataframe is equ
assert len(df_WB)==j[0]['total'], "DataFrame observatic
```

countryiso3code

countryiso3code
KEN
KEN
KEN
KEN
KEN

Data on Contraceptive prevalence: Any modern method (Percent) for Kenya can be

```

country_M49 = 404 #set the country M49 code
indicator_code_UNPD = 2 #set the indicator code
start_year = 1990 #set the start year
end_year = 2020 #set the end year


target = base_url_UNPD + f"/data/indicators/{indicator_

response = requests.get(target) #call the UNPD Data Por
j = response.json()
df_UNPD = pd.json_normalize(j['data'])

# As long as the response contains information in the '
while j['nextPage'] is not None:
    response = requests.get(j['nextPage'])
    j = response.json()
    df_temp = pd.json_normalize(j['data'])
    df_UNPD = df_UNPD.append(df_temp)

# Verifies that the number of records available from AP
assert len(df_UNPD)==j['total'], "DataFrame observatior

#Filter data to only include median estimates for all w
df_UNPD = df_UNPD.loc[(df_UNPD['variant']=='Median') &
```

locationId	location	iso3	iso2	locationTypeId	
404	Kenya	KEN	KE	4	2
404	Kenya	KEN	KE	4	2
404	Kenya	KEN	KE	4	2
404	Kenya	KEN	KE	4	2
404	Kenya	KEN	KE	4	2

The two datasets obtained through the APIs can now be merged together. The mat

```
#Rename columns for merging
df_UNPD = df_UNPD.rename(columns={"timeLabel":"year", '
df_WB = df_WB.rename(columns={"countryiso3code":"iso3",

# Merge dataframes from World Bank and UN Data Portal
df = pd.merge(df_UNPD[['location', 'iso3', 'year', 'CP'
# Convert year to int for plotting
df['year'] = df['year'].astype(int)
```

location
Kenya
Kenya
Kenya
Kenya
Kenya

UN Statistics Division SDG API

Another interesting possibility is to plot the UNPD data with the Sustainable Develo

```
base_url_SDG = "https://unstats.un.org/SDGAPI/v1/sdg/"
```

The Sustainable Development Goals are 17:

```
df_SDG_goals = pd.read_json(base_url_SDG + "Goal/List")
```

FIELD1	code	
1	1	End poverty in all its forms everywhere
2	2	End hunger, achieve food security and improved nutrition and promot
3	3	Ensure healthy lives and promote well-being for all at all ages
4	4	Ensure inclusive and equitable quality education and promote lifelong
5	5	Achieve gender equality and empower all women and girls
6	6	Ensure availability and sustainable management of water and sanitat

Within each goal, there are one or more targets to be achieved:

```
df_SDG_targets = pd.read_json(base_url_SDG + "Target/Li
```

FIELD1	goal	code	
1	1	1.1	By 2030, eradicate extreme poverty for all people everywhere, cu
2	1	1.2	By 2030, reduce at least by half the proportion of men, women a
3	1	1.3	Implement nationally appropriate social protection systems and
4	1	1.4	By 2030, ensure that all men and women, in particular the poor a
5	1	1.5	By 2030, build the resilience of the poor and those in vulnerable
6	1	1.a	Ensure significant mobilization of resources from a variety of sc

And for each target there is one or more indicators to monitor progress towards it.

```
df_SDG_indicators = pd.read_json(base_url_SDG + "Indica
```

FIELD1	goal	target	code	description
1	1	1.1	1.1.1	Proportion of the population living below the international p age, employment status and geographic location (urban/ru
2	1	1.2	1.2.1	Proportion of population living below the national poverty li
3	1	1.2	1.2.2	Proportion of men, women and children of all ages living in dimensions according to national definitions
4	1	1.3	1.3.1	Proportion of population covered by social protection floors distinguishing children, unemployed persons, older persons disabilities, pregnant women, newborns, work-injury victims the vulnerable
5	1	1.4	1.4.1	Proportion of population living in households with access t
6	1	1.4	1.4.2	Proportion of total adult population with secure tenure right legally recognized documentation, and (b) who perceive the secure, by sex and type of tenure

Example 3. Demand for family planning satisfied vs. Maternal mortality ratio - Afr

In this example, the data on demand for family planning satisfied for African countr

```
SSA_countries = list(df_aggregates_UNPD['id'][df_aggreg
SSA_countries = ",".join(SSA_countries)

#set the indicator code
indicator_code_SDG = "SH_STA_MORT"
#set the reference period
timePeriodStart = 2017

# define relative path for API call
relative_path = f"Series/Data?seriesCode={indicator_co
# set target
target = base_url_SDG + relative_path
# Call the UN SDG API
response = requests.get(target)
# Format response to JSON
j = response.json()
# Read JSON data to a dataframe
df_SDG = pd.json_normalize(j['data'])

# loop through the pages returned in the API response a
for i in range(2, j['totalPages']+1):
    target = base_url_SDG + relative_path + f"&page={i
    response = requests.get(target)
    j = response.json()
    df_temp = pd.json_normalize(j['data'])
    df_SDG = df_SDG.append(df_temp)

# verify that dataframe length matches total number of
assert len(df_SDG)==j['totalElements'], "DataFrame obse
```

FIELD1	series	seriesDescription	
1	SH_STA_MORT	Maternal mortality ratio	396

FIELD1	series	seriesDescription	
2	SH_STA_MORT	Maternal mortality ratio	396
3	SH_STA_MORT	Maternal mortality ratio	396
4	SH_STA_MORT	Maternal mortality ratio	396
5	SH_STA_MORT	Maternal mortality ratio	396
6	SH_STA_MORT	Maternal mortality ratio	396

Data on *Demand for family planning satisfied by any modern method (Percent)* are

```
#set the indicator code
indicator_code_UNPD = 8
#set the start year
start_year = 2017
#set the end year
end_year = 2017

# set path for API call
relative_path = f"/data/indicators/{indicator_code_UNPD
target = base_url_UNPD + relative_path
# Call the API
response = requests.get(target)
# Convert response to JSON
j = response.json()
# Read JSON data into dataframe
df_UNPD = pd.json_normalize(j['data'])

# As long as there is data in the nextPage field, conti
while j['nextPage'] is not None:
    response = requests.get(j['nextPage'])
    j = response.json()
    df_temp = pd.json_normalize(j['data'])
    df_UNPD = df_UNPD.append(df_temp)

# Verify that the number of rows in the dataframe is ec
assert len(df_UNPD)==j['total'], "DataFrame observatior

# Filter the dataframe to only include median variant f
df_UNPD = df_UNPD.loc[(df_UNPD['variant']=="Median") &
```

FIELD1	locationId	location	iso3	iso2
1	24	Angola	AGO	AO
2	72	Botswana	BWA	BW
3	108	Burundi	BDI	BI
4	120	Cameroon	CMR	CM
5	132	Cabo Verde	CPV	CV
6	140	Central African Republic	CAF	CF

The two datasets are merged together. As the time period is unique (*2017*), the ma

```
# renames columns for merging
df_UNPD = df_UNPD.rename(columns={"value":"SatFP"})
```

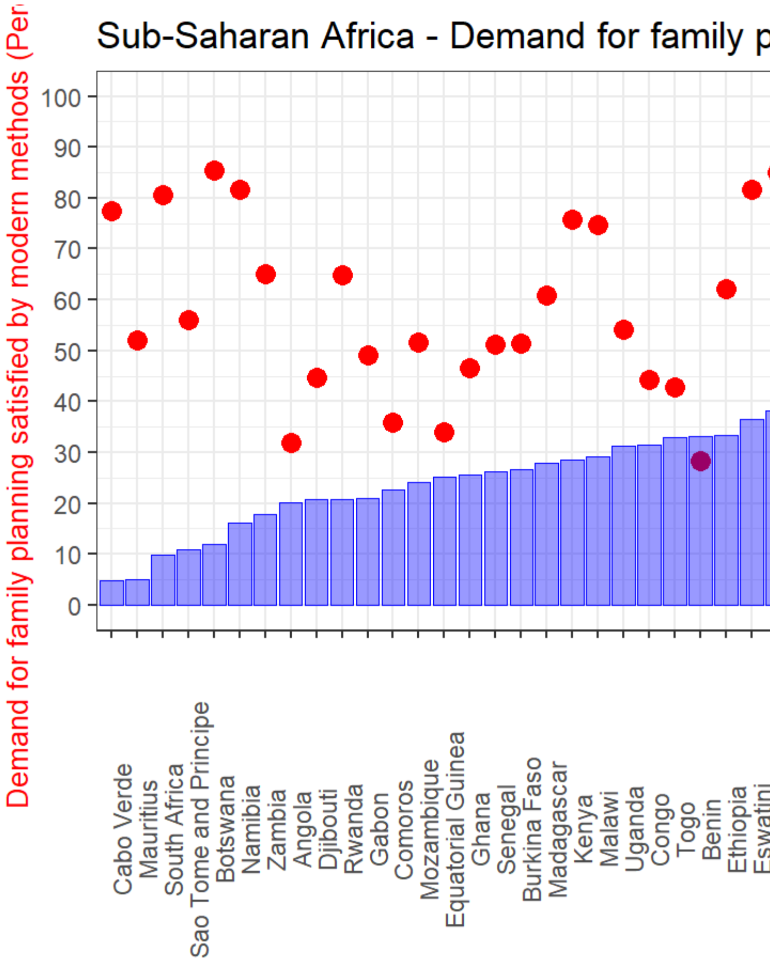
```
df_UNPD['locationId'] = df_UNPD['locationId'].astype(st
df_SDG = df_SDG.rename(columns={"geoAreaCode":"locatio
# Left join files on locationId
df = pd.merge(df_UNPD[['location', 'locationId', "SatF
```

```
df.loc[df['location']=="United Republic of Tanzania", '
df.loc[df['location']=="Democratic Republic of the Cong
```

	FIELD1
1	
2	
3	
4	
5	
6	

The plot below shows the results of the proportion of demand for family planning s

```
# Instantiate figure and axis objects
fig, ax = plt.subplots()

ax.scatter(df['location'], df['SatFP'], c='r') # draw s
ax.yaxis.set_ticks([i for i in range(0, 110, 10)]) # ad
ax.set_ylabel("Demand for family planning satisfied by
ax.set_xticklabels(df['location'], rotation=90) # rotat

ax2 = ax.twinx() # duplicate x axis
ax2.bar(df['location'], height=df['SDG_3.1.1'], alpha=0
ax2.yaxis.set_ticks([i for i in range(0, 1300, 100)]) #
ax2.set_ylabel("Maternal mortality ratio", color='b') #

ax.set_zorder(ax2.get_zorder()+1) #set the order in whi
ax.patch.set_visible(False)

fig.suptitle("Sub-Saharan Africa\nDemand for family pla
fig.subplots_adjust(top=0.8) # adjust spacing to fit ti
```



Example 4. Total fertility rates vs. female labour force participation

In the fourth example, data on total fertility rates are plotted together with data on f

```
#set base path for ILO
base_url_ILO = "https://www.ilo.org/data-api/rest/v1/da
#indicator
indicator_code_ILO = "DF_YI_ALL_EAP_DWAP_SEX_AGE_RT"
#set the start year
startPeriod = 2018
#set the collection, reference area, frequency and meas
collection_ILO, ref_area_ILO, frequency_ILO, measure_Il
#set the sex
sex_ILO = "SEX_F."
#set the age
age_ILO = "AGE_AGGREGATE_TOTAL"

#define the target url
target = base_url_ILO + f"{indicator_code_ILO}/{collect

#call the API
response = requests.get(target)
j = response.json()
df_ILO = pd.json_normalize(j)
```

FIELD1	collection	collection_Label	refarea	refarea_Label	indicator

FIELD1	collection	collection_Label	refarea	refarea_Label	indicator
1	YI	Annual indicators	ALB	Albania	EAP_DWAP_SEX_AGE_
2	YI	Annual indicators	ARE	United Arab Emirates	EAP_DWAP_SEX_AGE_
3	YI	Annual indicators	ARG	Argentina	EAP_DWAP_SEX_AGE_

The *Total Fertility Rate* can be accessed through the Data Portal API. As usual, it is

```python
#set the indicator code
indicator_code_UNPD = 19
#set the start year
start_year =  2018
#set the end year
end_year =  2018

# Get list of countries from geoarea dataframe and conv
countries = list(df_geoarea_UNPD['id'][df_geoarea_UNPD
countries = ",".join(countries)

#set target
target = base_url_UNPD + f"/data/indicators/{indicator_
# call the API
response = requests.get(target)
# convert response to JSON
j = response.json()
# Read JSON into dataframe
df_UNPD = pd.json_normalize(j['data'])

# As long as there is data in the nextPage field, conti
while j['nextPage'] is not None:
    response = requests.get(j['nextPage'])
    j = response.json()
    df_temp = pd.json_normalize(j['data'])
    df_UNPD = df_UNPD.append(df_temp)

# Verify that the length of the DataFrame is equal to t
assert len(df_UNPD)==j['total'], "DataFrame observation

# Filter data to only include median estimates
df_UNPD = df_UNPD.loc[(df_UNPD['variant']=="Median")]
```

FIELD1	locationId	location	iso3
1	4	Afghanistan	AFG
2	8	Albania	ALB

FIELD1	locationId	location	iso3	
3	12	Algeria	DZA	D
4	24	Angola	AGO	A
5	28	Antigua and Barbuda	ATG	A
6	31	Azerbaijan	AZE	A

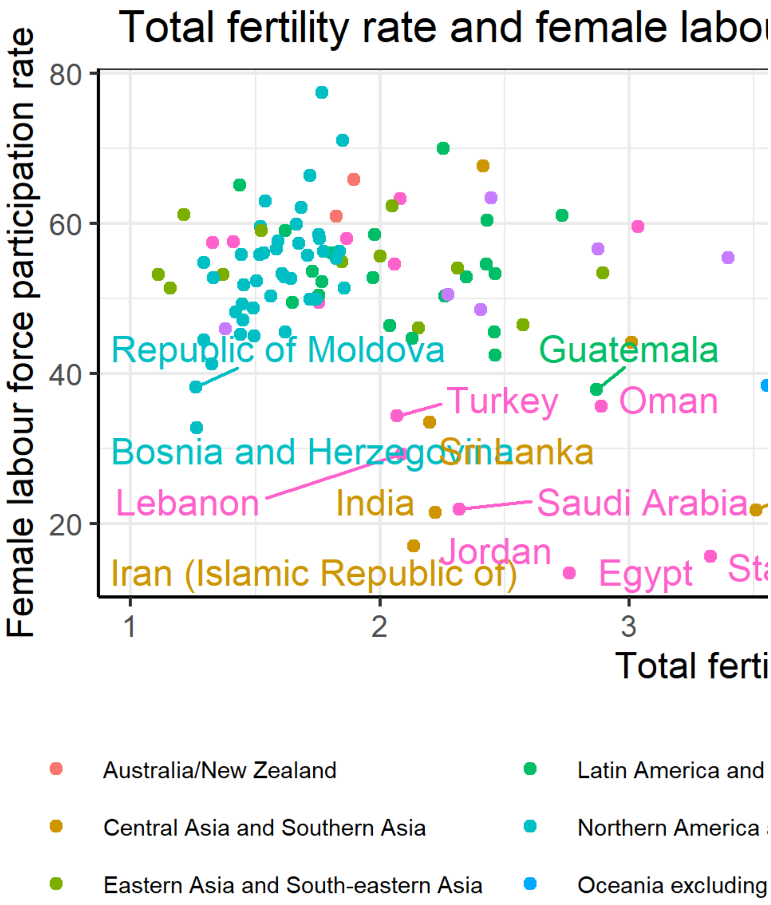
Merge the newly created dataset with the table with the SDG regional aggregates.

```
# Rename columns for merge
df_UNPD = df_UNPD.rename(columns={"value":"TFR"})
df_ILO = df_ILO.rename(columns={"refarea":"iso3", "obs_
# Left join dataframes from ILO and UNPD
df = pd.merge(df_UNPD[['location', "locationId", "iso3'
# Remove NaN values
df = df[~df['LabourParticipation'].isna()]

# Left join SDG region names from aggregates dataframe
df = pd.merge(df, df_aggregates_UNPD.loc[df_aggregates_
# rename column to SDGregion from parentName
df = df.rename(columns={"parentName":"SDGregion"})
```

FIELD1
1
2
3
4
5
6

The data are presented below as a scatter plot, with the data points distinguished b

```
# Generate list of unique values for SDG regions
sdg_regions = list(df['SDGregion'].unique())
# Create list of predefined colors
colors = ["#B33E52", "#FF7F0E", "#FFEE33", "#AAF400", '
# Create a dictionary of regions matched with colors ar
color_dict = dict(zip(sdg_regions, colors))
df['color'] = df['SDGregion'].map(color_dict)

# Generate list of country labels for countries with a
country_labels = list(df['location'][df['LabourParticip

# Instantiate figure and axis objects
fig,ax = plt.subplots()
for i in sdg_regions: # loop through regions and plot e
    ax.scatter(df.loc[df['SDGregion']==i,'TFR'], df.loc
ax.set_ylabel("Female labour force participation rate")
ax.set_xlabel("Total fertility rate") # set x-axis labe
ax.legend(title="SDG Region", loc="upper center", bbox_
for i in country_labels: #loop through selected countri
    ax.annotate(i, (df.loc[df['location']==i, "TFR"], c
fig.suptitle("Total fertility rate and female participa
```



FIELD1	
1	Albania
2	Azerbaijan
3	Argentina
4	Australia
5	Austria
6	Armenia

Tutorial using R

Basic Examples

In this section, some examples will be provided to show how to access the Data Pc

Example 1: Returning a list of indicators

In the first example, a call will be made to the API to return a complete list of indica

```
library(jsonlite)
library(httr)

# Declares the base url for calling the API
base_url <- "https://population.un.org/dataportalap

```

```
# Creates the target URL, indicators, in this insta
target <- paste0(base_url, "/indicators/")

# Get the response, which includes data as well as
response <- fromJSON(target)

# Get the first page of data
df <- response$data

# Loop until there are new pages with data
while (!is.null(response$nextPage)){

#call the API for the next page
response <- fromJSON(response$nextPage)

#add the data of the new page to the data.frame wi
df <- rbind(df, response$data)

}
```

To view response code:

```
status_code(GET(target))

## [1] 200
```

Example 2: Returning a list of geographical areas

In this example, the list of geographical areas are retrieved from the API:

```
# Update relative path to retrieve records on locat
target <- paste0(base_url, "/locations/")

# Call the API
response <- fromJSON(target)

# Get the first page of data
df <- response$data

# Get the other pages with data
while (!is.null(response$nextPage)){

    response <- fromJSON(response$nextPage)
    df <- rbind(df, response$data)

}

```

Example 3: Returning a single indicator for a single geographical area

The following example calls the API to return the contraceptive prevalence rate (inc

```
# Update the relative path to search for data on a
target <- paste0(base_url, "/data/indicators/1/loca

# Call the API
response <- fromJSON(target)

# Get the first page of data
df <- response$data

```

```
# Get the other pages with data
while (!is.null(response$nextPage)){

    response <- fromJSON(response$nextPage)
    df <- rbind(df, response$data)

}
```

And to view only data for all women and the median estimates:

```
df2 <- df[(df$variant=="Median") & df$category=="A
```

Example 4: Returning data on multiple indicators and geographical areas

Below is one more example using a more complicated search in which a user wish

```
callAPI <- function(relative_path, topics_list=FALS
    base_url <- "https://population.un.org/dataport
    target <- paste0(base_url, relative_path)
    response <- fromJSON(target)
    # Checks if response was a flat file or a list
    # If response is a list, we may need to loop th
    if (class(response)=="list"){
        # Create a dataframe from the first page of
        df <- response$data
        while (!is.null(response$nextPage)){
            response <- fromJSON(response$nextPage)
            df_temp <- response$data
            df <- rbind(df, df_temp)
        }
        return(df)}
    # Otherwise, we will simply load the data direc
    else{
        if (topics_list==TRUE){
            df <- fromJSON(target, flatten = TRUE)
            return(df[[5]][[1]])
        }
        else{
            df <- fromJSON(target)
            return(df)
        }
    }
}

# Uses callAPI function to get a list of locations
df_locations <- callAPI("/locations/")

# Identifies ID code for Western Africa
western_africa_id <- df_locations[df_locations$name

# Restricts the dataframe to only include geographi
country_codes <- as.character(df_locations[df_locat
country_codes <- paste(country_codes, collapse = ",

# Uses callAPI function to get a list of only Famil
df_topics <- callAPI("/topics/FP/indicators", topic
indicator_codes <-as.character(df_topics$id)
indicator_codes <- paste(indicator_codes, collapse

target <- paste0("/data/indicators/",indicator_code

df <- callAPI(target)
```

```

df2 <- df[(df$variant=="Median") & (df$category=="A
```

Output restricted to median values for all women and first ten rows:

Detailed Examples

This section provides some examples in R of how to access demographic indicato

Input

As first step, it is necessary to upload all the R packages needed to run the script.

```
library(jsonlite)
library(httr)
library(dplyr)
library(ggplot2)
library(ggrepel)
library(data.table)
```

UNPD Data Portal

The data included in the Data Portal can be accessed using the following base URL

```
base_url_UNPD <- "https://population.un.org/datapo
```

The database includes 9 topics:

```
# Define the target url
target <- paste0(base_url_UNPD, "/topics/")

# Call the API
response <- fromJSON(target)

# Get the data
df_topics_UNPD <- response$data
```

id	
0	Not applicabl
1	Population
2	Fertility
3	Mortality
4	International
5	Family Planni
6	Marital Statu
7	All Compone
8	Child Mortali
9	Maternal Mo

And covers the following list of geographical areas:

```
# Define the target url
target <- paste0(base_url_UNPD, "/locations/")

# Call the API
response <- fromJSON(target)

# Get the data
df_geoarea_UNPD <- response$data

# Get the other pages with data
while (!is.null(response$nextPage)){

    response <- fromJSON(response$nextPage)
    df_geoarea_UNPD <- rbind(df_geoarea_UNPD, respo

}
```

id	parentId
4	5501
8	925
12	912
16	957
20	925

For each geographical area, there are different groupings and aggregates (SDG Reg

```
# Define the target url
target <- paste0(base_url_UNPD, "/locationsWithAggr

# Get the data
df_aggregates_UNPD <- fromJSON(target)


```

FIELD1	id	
1	4	Afghanistan
2	4	Afghanistan
3	4	Afghanistan
4	4	Afghanistan
5	4	Afghanistan
6	8	Albania

World Bank Open Data

The first examples involve the use of the data queried through the World Bank Ope

```
base_url_WB <- "http://api.worldbank.org/v2/"
```

The datasets are organized by topics:

```
# Define the target url
target <- paste0(base_url_WB, "topic?format=json")

# Call the API
response <- fromJSON(target)

# Get the data
df_WB <- response[[2]]
```

id	value	
1	Agriculture & Rural Development	For the 70 percent of the world's poor who l
2	Aid Effectiveness	Aid effectiveness is the impact that aid has
3	Economy & Growth	Economic growth is central to economic de
4	Education	Education is one of the most powerful instr
5	Energy & Mining	The world economy needs ever-increasing a

Within each topic there are multiple indicators. The following code loads the indica

```
# Define the target url
target <- paste0(base_url_WB, "topic/11/indicator?f

# Call the API
response <- fromJSON(target)

#Get the data
df_WB <- response[[2]]
```

Here is a look at an extract of the possible indicators included:

id	value	
1	Agriculture & Rural Development	For the 70 percent of the world's poor who l
2	Aid Effectiveness	Aid effectiveness is the impact that aid has
3	Economy & Growth	Economic growth is central to economic de
4	Education	Education is one of the most powerful instr
5	Energy & Mining	The world economy needs ever-increasing a

Example 1. Adolescent fertility vs. net enrollment in primary school, Senegal

This example illustrates an exploratory comparison between the trends of the adol

```
#set the desired country
country_iso3 <- "SEN"
#set the  desired indicator
indicator_code_WB <- "SE.PRM.NENR"

#define the target url
target <- paste0(base_url_WB, "country/", country_i
#basic data query
response <- fromJSON(target)
```

```
#total pages with data available
Pages <- response[[1]]$pages

#create a list where to store the tables of data ob
df_WB <- list()

#for each page with data available
for (page in seq(1, Pages)){

    #get the url for the selected page
    target <-paste0(base_url_WB, "country/", countr
                    "?page=", page, "&format=json")

    #call the API
    response<- fromJSON(target)
    #the table with the data for each page is inclu
    df_WB[[page]] <- response[[2]]
}

#combine the various pages in a data.frame
df_WB <- rbind_pages(df_WB)
```

countryiso3code
SEN
SEN
SEN
SEN
SEN

The data on fertility rates for women aged 15-19, instead, can be accessed through

```
#set the country M49 code
country_M49 <- 686
#set the indicator code
indicator_code_UNPD <- 17
#set the start year
start_year <-  1970
#set the end year
end_year <-  2020

#define the target URL
target <- paste0(base_url_UNPD, "/data/indicators/"
                 "/locations/", country_M49, "/sta

#call the API
response <- fromJSON(target)

#get the table with data available in the first pag
df_UNPD <- response$data

#until there are next pages available
while (!is.null(response$nextPage)){
    #call the API for the next page
    response <- fromJSON(response$nextPage)
    #add the data of the new page to the data.frame
    df_UNPD <- rbind(df_UNPD, response$data)
}

```

```
#select the median variant, the age start 15 and th
df_UNPD <- df_UNPD %>%
  dplyr::filter(variant == "Median",
                ageStart == 15,
                ageEnd == 19)
```

locationId	location	iso3	iso2	locationTypeId	
686	Senegal	SEN	SN	4	1
686	Senegal	SEN	SN	4	1
686	Senegal	SEN	SN	4	1
686	Senegal	SEN	SN	4	1
686	Senegal	SEN	SN	4	1

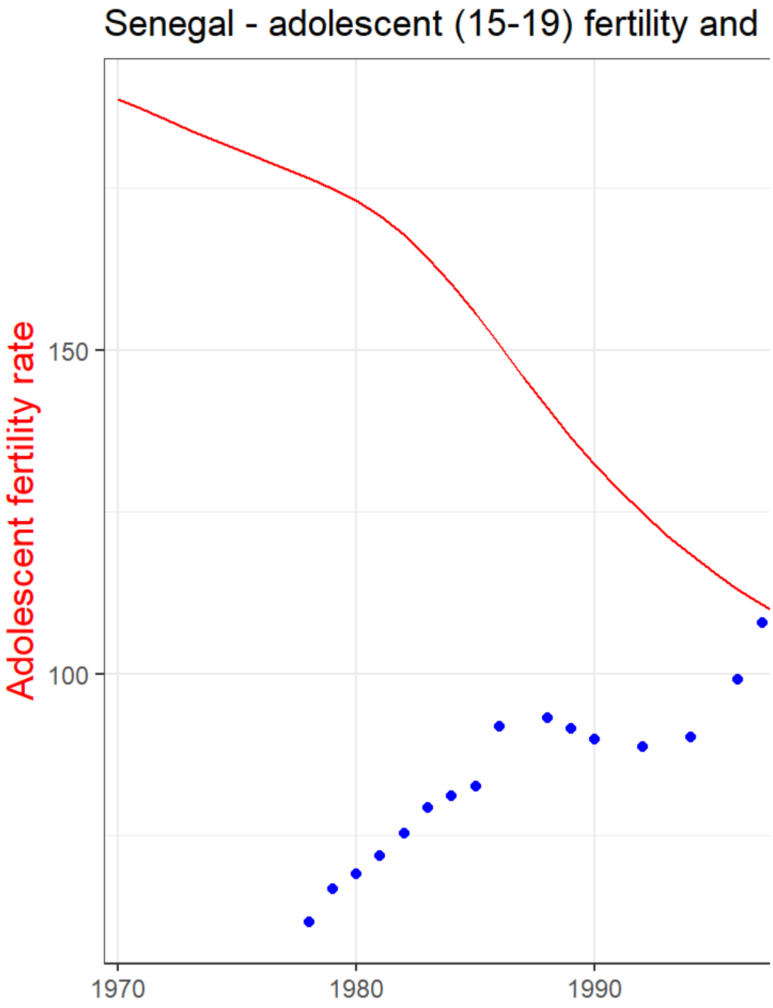
The two datasets obtained through the APIs are merged together. The matching va

```
#merge the two dataset
df <- df_UNPD %>%
  #from the ASFR 15-19 dataset keep only the variab
  dplyr::select(location, iso3, timeLabel, value) %
  dplyr::rename(year = timeLabel, #rename the time
                ASFR= value) %>%  #rename the value
  left_join(df_WB %>%
              #from the enrollment dataset keep onl
              dplyr::select(countryiso3code, date,
              dplyr::rename(iso3 = countryiso3code,
                            year = date, #rename th
                            Enrollment = value), #r
            by = c("iso3", "year")) #merge the two
```

FIELD1
42
43
44
45
46
47
48
49
50
51

Now it is possible to plot the two indicators together to explore their trends over tin

```
fig <- ggplot(df %>%
         dplyr::filter(year >= 1970), #filter only
       aes(x = year, group = 1)) + #the time is on
  geom_line(aes(y = ASFR), color = "red") + #the AS
  #the enrollment is on the second y-axis, so it is
  geom_point(aes(y = Enrollment*2), color = "blue")
  scale_y_continuous("Adolescent fertility rate", #
    sec.axis = sec_axis(~. /2, #scale the second y-
                        name="Net enrollment in pri
  theme_bw() + #set the theme as black and white
  theme(
    #set color and the size of the title of the fir
    axis.title.y = element_text(color = "red", size
    #set the color and the size of the title of the
    axis.title.y.right = element_text(color = "blue
    axis.title.x = element_blank(), #remove the tit
    plot.title = element_text(hjust = 0.5)) + #cent
  #set the title of the plot
  ggtitle("Senegal - Adolescent (15-19) fertility a
  scale_x_discrete(breaks=seq(1970, 2020, by = 10))
```



Example 2. Modern contraceptive prevalence and GDP per capita, Kenya

This example uses the data on modern contraceptive prevalence and on GDP per c

```
#set the ISO 3 code for the desired country
country_iso3 <- "KEN"
#set the code for the desired indicator
indicator_code_WB <- "NY.GDP.PCAP.CD"

#define the target url
target <- paste0(base_url_WB, "country/", country_i
#basic data query
response <- fromJSON(target)
```

```
#total pages
Pages <- response[[1]]$pages

#create a list where to store the tables of data ob
df_WB <- list()

for (page in seq(1, Pages)){

    #get the url for the selected page
    target <-paste0(base_url_WB, "country/", countr

    #the data frame for each page is included in th
    df_WB[[page]] <- fromJSON(target)[[2]]
}

#combine the various pages in a data.frame
df_WB <- rbind_pages(df_WB)
```

countryiso3code
KEN
KEN
KEN
KEN
KEN

Data on *Contraceptive prevalence: Any modern method (Percent)* for Kenya can be

```
#set the country M49 code
country_M49 <- 404
#set the indicator code
indicator_code_UNPD <- 2
#set the start year
start_year <-  1990
#set the end year
end_year <-  2020

#define the target URL
target <- paste0(base_url_UNPD, "/data/indicators/'
                 "/start/", start_year, "/end/", er

#call the API
response <- fromJSON(target)

#get the table with data available in the first pag
df_UNPD <- response$data

#until there are next pages available
while (!is.null(response$nextPage)){

  #call the API for the next page
  response <- fromJSON(response$nextPage)

  #add the data of the new page to the data.frame w
  df_UNPD <- rbind(df_UNPD, response$data)
}

#select the median variant and the all women catego
df_UNPD <- df_UNPD %>%
  dplyr::filter(variant == "Median",
                category == "All women")
```

locationId	location	iso3	iso2	locationTypeId	
404	Kenya	KEN	KE	4	2
404	Kenya	KEN	KE	4	2
404	Kenya	KEN	KE	4	2
404	Kenya	KEN	KE	4	2
404	Kenya	KEN	KE	4	2

The two datasets obtained through the APIs are merged together. The matching va

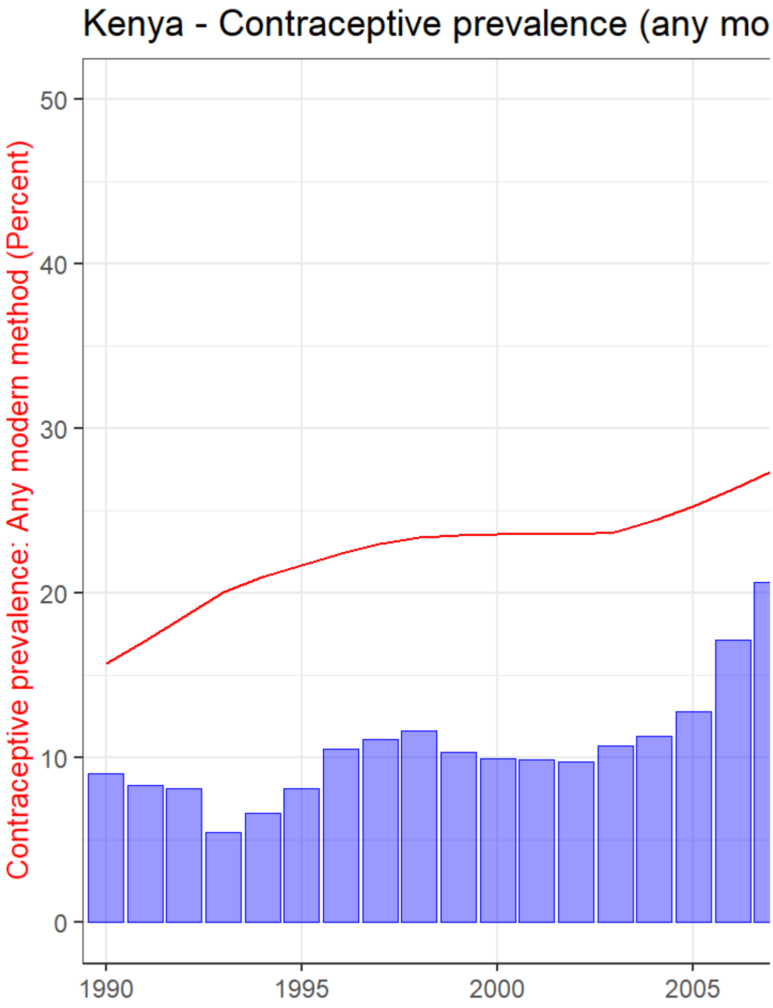
```
#merge the two datasets
df <- df_UNPD %>%
  #from the CP dataset keep only the variables need
  dplyr::select(location, iso3, timeLabel, value) %
  dplyr::rename(year = timeLabel, #rename the time
                CP = value) %>%   #rename the value
  left_join(df_WB %>%
              #from the GDP keep only the variables
              dplyr::select(countryiso3code, date,
              dplyr::rename(iso3 = countryiso3code,
                            year = date, #rename th
                            GDP = value), #rename t
            by = c("iso3", "year")) #merge the two
```

location
Kenya
Kenya
Kenya
Kenya
Kenya

Finally, it is possible to display the two indicators together: the GDP per capita as b

```
fig <- ggplot(df, aes(x = year, group = 1)) + #the
  geom_line(aes(y = CP), color = "red") + #the CP i
  #the GDP is on the second y-axis, so it is divide
  geom_bar(aes(y = GDP/40), stat="identity", size=.
  scale_y_continuous("Contraceptive prevalence: Any
    sec.axis = sec_axis(~. *40,   #scale the second
                        name="GDP per capita (curre
                        breaks = seq(0,2000, by=500
    limits = c(0,50), #set the limits of the first
    breaks = seq(0,50, by=10)) + #set the labels to
  theme_bw() + #set the theme as black and white
  theme(
    #set the color and the size of the title of the
    axis.title.y = element_text(color = "red", size
    #set the color and the size of the title of the
    axis.title.y.right = element_text(color = "blue
    axis.title.x = element_blank(), #remove the tit
```

```
        plot.title = element_text(hjust = 0.5)) + #cent
      #set the title of the plot
      ggtitle("Kenya - Contraceptive prevalence (any mo
      scale_x_discrete(breaks=seq(1990, 2020, by = 5))
```



UN Statistics Division SDG API

Another interesting possibility is to plot the UNPD data with the Sustainable Develo

```
base_url_SDG <- "https://unstats.un.org/SDGAPI/v1/s
```

The Sustainable Development Goals are 17:

```
df_SDG_goals <- fromJSON(paste0(base_url_SDG, "Goal
```

FIELD1	code	
1	1	End poverty in all its forms everywhere
2	2	End hunger, achieve food security and improved nutrition and promot
3	3	Ensure healthy lives and promote well-being for all at all ages
4	4	Ensure inclusive and equitable quality education and promote lifelong
5	5	Achieve gender equality and empower all women and girls
6	6	Ensure availability and sustainable management of water and sanitat

Within each goal, there are one or more targets to be achieved:

```
df_SDG_target <- fromJSON(paste0(base_url_SDG, "Ta
```

FIELD1	goal	code	
1	1	1.1	By 2030, eradicate extreme poverty for all people everywhere, cu
2	1	1.2	By 2030, reduce at least by half the proportion of men, women a
3	1	1.3	Implement nationally appropriate social protection systems and
4	1	1.4	By 2030, ensure that all men and women, in particular the poor a
5	1	1.5	By 2030, build the resilience of the poor and those in vulnerable
6	1	1.a	Ensure significant mobilization of resources from a variety of sc

And for each target there is one or more indicators to monitor progress towards it.

```
df_SDG_indicators <- fromJSON(paste0(base_url_SDG,
```

FIELD1	code	
1	1	End poverty in all its forms everywhere
2	2	End hunger, achieve food security and improved nutrition and promot
3	3	Ensure healthy lives and promote well-being for all at all ages
4	4	Ensure inclusive and equitable quality education and promote lifelong
5	5	Achieve gender equality and empower all women and girls
6	6	Ensure availability and sustainable management of water and sanitat

Example 3. Demand for family planning satisfied vs. Maternal mortality ratio - Afr

In this example, the data on demand for family planning satisfied for African countr

```
#create a vector with the M49 codes of the Sub-Saha
SSA_countries <- df_aggregates_UNPD %>%
  dplyr::filter(SDGRegion == "Sub-Saharan Africa")
  pull(Id)
#set the indicator code
indicator_code_SDG <- "SH_STA_MORT"
#set the reference period
timePeriodStart <- 2017

#query the data
df_SDG <- lapply(SSA_countries, function(country){

  #define the target url
  target <- paste0(base_url_SDG, "Series/Data?serie
                   "&timePeriodStart=", timePeriodS
  #call the API
  response <- fromJSON(target)
  #return the item "data" as a data.table object
  return(data.table(response$data, keep.rownames =
})
```

```

# create the final data frame
df_SDG <- rbind_pages(df_SDG)
```

FIELD1	series	seriesDescription	
1	SH_STA_MORT	Maternal mortality ratio	396
2	SH_STA_MORT	Maternal mortality ratio	396
3	SH_STA_MORT	Maternal mortality ratio	396
4	SH_STA_MORT	Maternal mortality ratio	396
5	SH_STA_MORT	Maternal mortality ratio	396
6	SH_STA_MORT	Maternal mortality ratio	396

Data on *Demand for family planning satisfied by any modern method (Percent)* are

```
#set the indicator code
indicator_code_UNPD <- 8
#set the start year
start_year <-  2017
#set the end year
end_year <-  2017

#query the data
df_UNPD <-  lapply(SSA_countries, function(country)

  #define the target url
  target <- paste0(base_url_UNPD, "/data/indicators
                          "/locations/", country, '

  #this loop will allow to skip to the next country
  for(j in country){
    #define skip next as false
    skip_to_next <- FALSE

    tryCatch({
      #call the API
      response <- fromJSON(target)
      #get the table with data available
      df <-  response$data
      return(df)
      },

      error = function(e) { skip_to_next <<- TRUE})

    if(skip_to_next) { next }

  }
})

#select the median variant and the all women catego
df_UNPD <- rbind_pages(df_UNPD) %>%
  dplyr::filter(variant == "Median",
                category == "All women")
```

FIELD1	locationId	location	iso3	iso2
1	24	Angola	AGO	AO

FIELD1	locationId	location	iso3	iso2
2	72	Botswana	BWA	BW
3	108	Burundi	BDI	BI
4	120	Cameroon	CMR	CM
5	132	Cabo Verde	CPV	CV
6	140	Central African Republic	CAF	CF

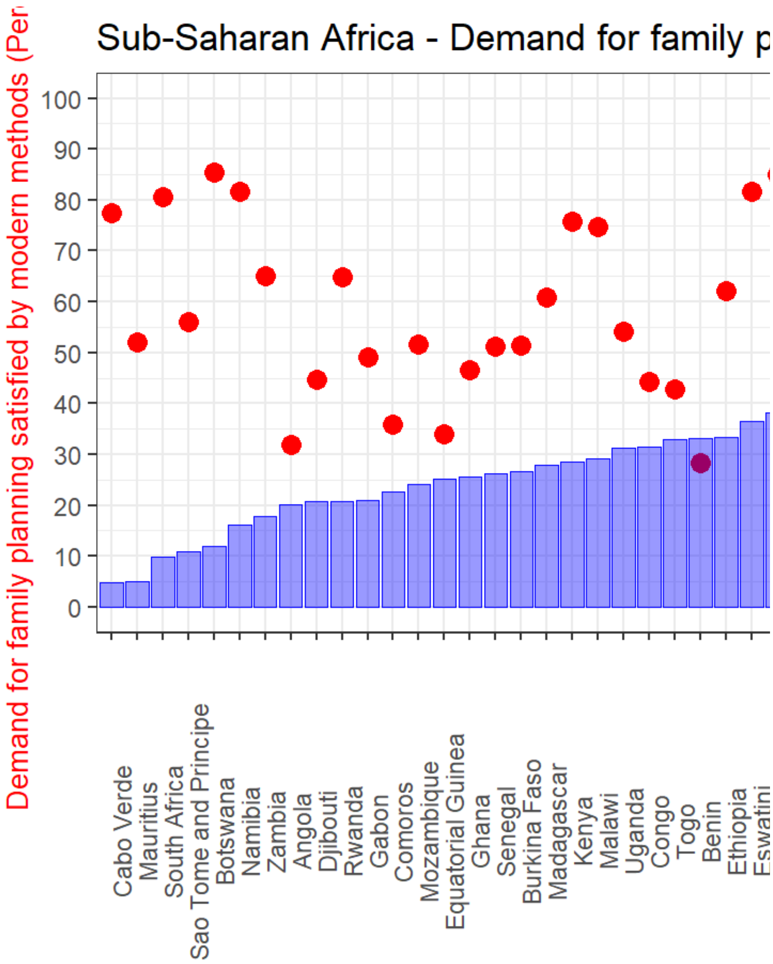
The two datasets are merged together. As the time period is unique (*2017*), the ma

```
#merge the two datasets
df <- df_UNPD %>%
  dplyr::select(location, locationId, value) %>% #f
  dplyr::rename(SatFP = value) %>%  #rename the val
  mutate(locationId = as.character(locationId)) %>%
  left_join(df_SDG %>%
              dplyr::select(geoAreaCode, value) %>%
              dplyr::rename(locationId = geoAreaCod
                            SDG_3.1.1 = value), #re
            by = c("locationId")) %>% #merge the tw
  dplyr::filter(!is.na(SDG_3.1.1)) %>% #remove the
  mutate(SDG_3.1.1 = as.numeric(SDG_3.1.1)) #transf
```

FIELD1
1
2
3
4
5
6

The plot below shows the results of the proportion of demand for family planning s

```
fig <- ggplot(df,
  #the locations are on the x-axis, and need to be
  aes(x = reorder(location, SDG_3.1.1), group = 1)
  geom_point(aes(y = SatFP), color = "red", size =
  geom_bar(aes(y = SDG_3.1.1/12), #the SDG 3.1.1 is
           stat="identity", size=.1, color = "blue
  #set the title of the first y-axis
  scale_y_continuous("Demand for family planning sa
    sec.axis = sec_axis(~. *12,  #scale the second
                        name="Maternal mortality ra
                        breaks = seq(0,1200, by=200
    limits = c(0,100), #set the limits of the first
    breaks = seq(0,100, by=10)) + #set the labels
  theme_bw() + #set the theme as black and white
  theme(
    #set the color and the size of title of the fir
    axis.title.y = element_text(color = "red", size
    #set the color and the size of title of the sec
    axis.title.y.right = element_text(color = "blue
    axis.title.x = element_blank(), #remove the tit
    #set the orientation and the size of the label
    axis.text.x = element_text(angle = 90, size = 7
    plot.title = element_text(hjust = 0.5)) + #cent
  #set the title of the plot
  ggtitle("Sub-Saharan Africa - Demand for family p
          and maternal mortality ratio - 2017")
```



Example 4. Total fertility rates vs. female labour force participation

In the fourth example, data on total fertility rates are plotted together with data on f

```
    #indicator
    indicator_code_ILO <- "DF_YI_ALL_EAP_DWAP_SEX_AGE_R
    #set the start year
    startPeriod <-  2018
    #set the collection, reference area, frequency and
    collection_ILO = ref_area_ILO = frequency_ILO = mea
    #set the sex
    sex_ILO <- "SEX_F."
    #set the age
    age_ILO <- "AGE_AGGREGATE_TOTAL"

    #define the target url
    target <- paste0("https://www.ilo.org/data-api/rest
                     ref_area_ILO, frequency_ILO, measu
                     startPeriod, "&LASTNOBSERVATIONS=1
    #call the API
    df_ILO <- fromJSON(target)
```

FIELD1	collection	collection_Label	refarea	refarea_Label	indicator
1	YI	Annual indicators	ALB	Albania	EAP_DWAP_SEX_AGE_
2	YI	Annual indicators	ARE	United Arab Emirates	EAP_DWAP_SEX_AGE_
3	YI	Annual indicators	ARG	Argentina	EAP_DWAP_SEX_AGE_

The *Total Fertility Rate* can be accessed through the Data Portal API. As usual, it is

```
#set the indicator code
indicator_code_UNPD <- 19
#set the start year
start_year <-  2018
#set the end year
end_year <-  2018

#query the data
df_UNPD <-  lapply(df_geoarea_UNPD$id, function(cou

  #define the target url
  target <- paste0(base_url_UNPD, "/data/indicators
                          "/locations/", country, '

  #this loop will allow to skip to the next country
  for(j in country){
    #define skip next as false
    skip_to_next <- FALSE

    tryCatch({
      #call the API
      response <- fromJSON(target)
      #get the table with data available
      df <-  response$data
      return(df)
      },

      error = function(e) { skip_to_next <<- TRUE})

    if(skip_to_next) { next }

  }
})

# select only the median variant and the areas that
```

```
df_UNPD <- rbind_pages(df_UNPD) %>%
  dplyr::filter(variant == "Median",
                locationTypeId == 4)
```

FIELD1	locationId	location	iso3
1	4	Afghanistan	AFG
2	8	Albania	ALB
3	12	Algeria	DZA
4	24	Angola	AGO
5	28	Antigua and Barbuda	ATG
6	31	Azerbaijan	AZE

The two datasets are merged together using as matching variable the ISO3 country

```
#merge the datasets together
df <- df_UNPD %>%
  dplyr::select(location, locationId, iso3, value)
  dplyr::rename(TFR = value) %>% #rename the values
  left_join(df_ILO %>%
              dplyr::select(refarea,  obs_Value) %>
              dplyr::rename(iso3 = refarea, #rename
                            LabourPartipation = obs
            by = c("iso3")) %>% #merge the two data
  dplyr::filter(!is.na(LabourPartipation)) #remove
```

FIELD1
1
2
3
4
5
6

Merge the newly created dataset with the table with the SDG regional aggregates.

```
#merge the newly created dataset with the table of
df <- df %>%
  left_join(df_aggregates_UNPD %>%
              dplyr::filter(parentTypeName == "SDG
              dplyr::select(iso3, parentName) %>% #
              dplyr::rename(SDGRegion = parentName)
            by = "iso3") #merge the two dataset usi
```

FIELD1	
1	Albania
2	Azerbaijan

FIELD1	
3	Argentina
4	Australia
5	Austria
6	Armenia