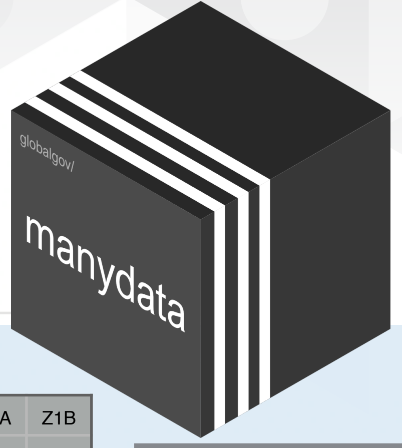
Explore the data with manydata: : CHEAT SHEET

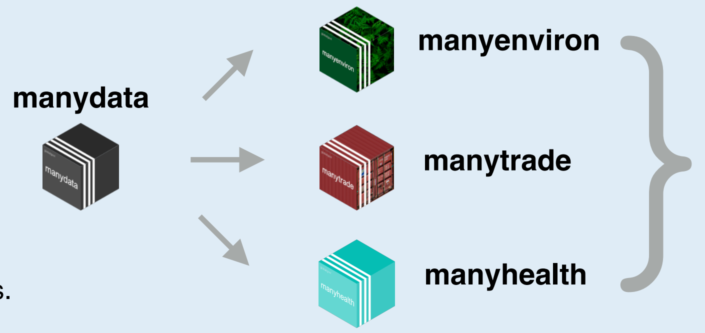
manydata is the portal to packages that include many datasets to different domains of global governance. Using the functions in **manydata,** users can call, compare, and consolidate different datasets and datacubes across various domains of global governance.

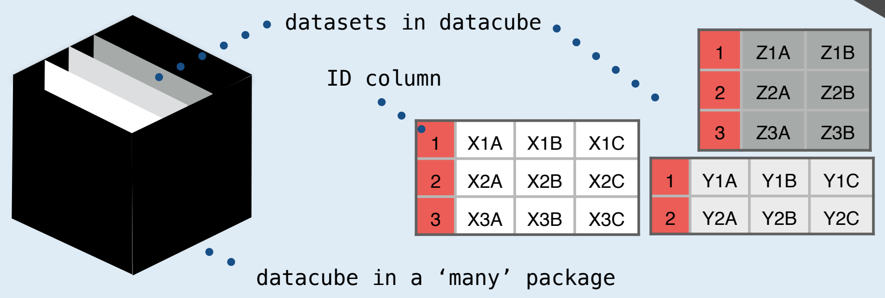


1) Call

```
call_packages(manypackage, develop)
```

call_packages() is a quick and easy way to access and install 'many' packages. The function allows users to interactively select the 'develop' branch using the '**develop**' argument. Running the function without an argument returns the full list of 'many' packages.





Datasets in a datacube have potentially overlapping IDs, overlapping rows/ observations, and overlapping columns/ variables that may have the same or different values

```
call_treaties(dataset, treaty_type, variable, actor)
```

Use '**treaty_type**', '**variable**', and '**actor**' arguments to extract the relevant observations for specific treaties ("bilateral" or "multilateral"), variables, or actors in the dataset.

manyID	stateID	Title	Begin
TFJXKC_1999O	BRA	B	1999-02-28
ECH_2003A	FRA	M	2003-07-13
AGEJKL_1947O	KEN	A	1947-09-19
BALTTT_1966O	NZL	T	1966-05-08

```
treaty_type = "bilateral",
variable = c("Title", "Begin")
```

manyID	stateID1	stateID2	Title	Begin
TFJXKC_1999O	SIN	BRA	B	1999-02-28
BALTTT_1966O	NZL	MEX	T	1966-05-08

```
call_sources(manypackage, datacube,
dataset, open_script, open_codebook)
```

call_sources() returns a tibble of sources ('Source', 'URL') and renamed variables ('Mapping') for each dataset in a datacube of a 'many' package.

Dataset	Source	URL	Mapping
Dataset_A	"Name Surname of authors, year, paper title using the data, publisher, place"	http…	from - to I Label - Title…
Dataset_B	"Name Surname of authors, year, paper title using the data, publisher, place"	http…	from - to I Treaty - Title…
Dataset_C	"Name Surname of authors, year, paper title using the data, publisher, place"	http…	from - to I Treaty - Title…

2) Compare

To identify the most suitable dataset(s) within a datacube for use, the '**compare_**' family of functions facilitates the comparison of observations within and across datasets in a datacube by various conditions:
- number/names of variables and observations
- range
- overlapping observations
- missing observations
- in categories ("confirmed", "majority", "unique", "missing", and "conflict")

Observations are matched by a 'key', usually an 'ID' variable like 'manyID' to facilitate comparison. Each unique state or treaty has a unique stateID or manyID that is the same across datasets. Results of comparisons are returned in a tibble. Each of these comparisons can be visualised using '**plot()**' on the output of '**compare_**' functions.

```
compare_dimensions(datacube, dataset)
```

compare_data() lists the observations, variables, and earliest and latest dates in each dataset in a datacube.

Dataset	Observations	Variables	Earliest_Date	Latest_Date
Dataset_A	70	15	1873-01-01	2020-12-20
Dataset_B	53	7	1986-03-05	2020-12-20
Dataset_C	96	5	1945-01-01	2022-01-01

```
compare_ranges(datacube, dataset, variable)
```

compare_ranges() returns the minimum, maximum, mean and median values for a specified variable.

Dataset	Variable	Min	Max	Mean	Median
Dataset_A	Begin	1873-01-01	2020-12-20	1946-12-27	1946-12-27
Dataset_B	Begin	1986-03-05	2020-12-20	2003-07-28	2003-07-29
Dataset_C	Begin	1945-01-01	2022-01-01	1983-07-03	1983-07-03

```
compare_overlap(datacube, dataset, key, variable, category)
```

Dataset 1

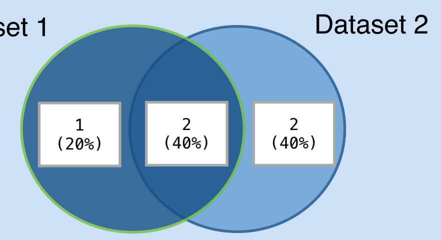
manyID	Begin
ABC	1995
ABC	1995
ABD	2001
BDF	2002

Dataset 2

manyID	Begin
ABC	1995
ABD	2001
BBC	1997
CFD	2003
CFD	2003
CFD	2003

Dataset	`Overlapping Observations`
Dataset 1	1
Dataset 2	2
Dataset1 .. Dataset 2	2

```
plot()
```



3) Consolidate

```
consolidate(datacube,
rows, cols, resolve, key)
```

consolidate() allows users to produce a single dataset from different datasets within the datacube by matching rows and resolving conflicts in data.

Datacubes are consolidated using '**key**', an identifying variable for each row (eg. "manyID"), to match rows across datasets. Select a method ("min", "max", "median", "mean", "coalesce", "random") to **resolve** conflicts among matched observations across datasets when consolidating.
For '**rows**' and '**cols**', enter either "any" to retain all rows/cols present across datasets or "every" to retain only rows/cols that appear in all datasets that are being consolidated.

manyID	DatasetA$Begin	DatasetB$Begin	DatasetC$Begin	DatasetD$Begin
ABC_1995O	1995-01-01	1995-03-04	1995-12-07	NA

```
resolve = "min"
```

manyID	Begin
ABCEFG_1995O	1995-01-01

```
resolve = "max"
```

manyID	Begin
ABCEFG_1995O	1995-12-07

```
resolve = "median"
```

manyID	Begin
ABCEFG_1995O	1995-03-04

```
resolve = "mean"
```

manyID	Beg
ABCEFG_1995O	1995-05-15

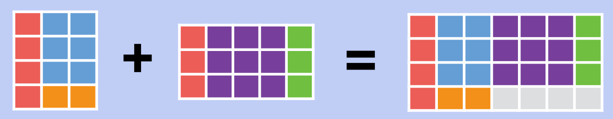
```
resolve = "random"
```

manyID	Begin
ABCEFG_1995O	1995-03-04

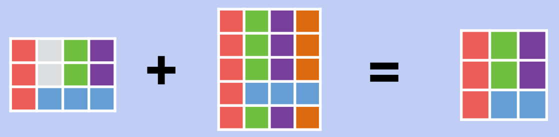
```
resolve = "coalesce"
```

manyID	Begin
ABCEFG_1995O	1995-01-01

```
rows & cols = "any"
```



```
rows & cols = "every"
```



Use **favour()** to specify the reference dataset for the first NA value before consolidating.