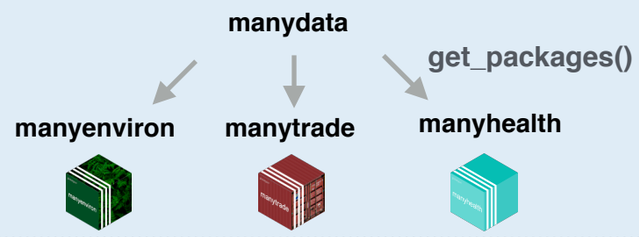
Explore the data with manydata: : CHEAT SHEET

manydata is the portal through which the data stored in the 'many' universe of packages can be easily accessed.
Using the functions in **manydata,** users can call, consolidate, and compare different datasets and databases across issue-domains of global governance.

1) Call

```
get_packages(manypackage)
```

get_packages() is a quick and easy way to access and download data from the many packages universe. The function allows users to interactively select which branch (main/develop).




```
data_source(manypackage, database, dataset)
```

Database from many packages universe

Dataset	Reference
Dataset_A	"Name Surname of authors, year, paper title using the data, publisher, place"
Dataset_B	"Name Surname of authors, year, paper title using the data, publisher, place"
Dataset_C	"Name Surname of authors, year, paper title using the data, publisher, place"

```
retrieve_bilaterals(dataset)
```

manyID	Title	Beg
TFJXKC_1999O	B	1999-02-28
ECE_2003A	M	2003-07-13
AGEJKL_1947O	A	1947-09-19
BALTTT_1966O	T	1966-05-08

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

```
retrieve_multilaterals(dataset)
```

manyID	Title	Beg
ECE_2003A	M	2003-07-13
AGEJKL_1947O	A	1947-09-19

retrieve_bilaterals() and **retrieve_multilaterals()** extract bilateral or multilateral treaties from a dataset of treaties.

```
data_contrast(manypackage, database, dataset)
```

Database

Dataset	Unique ID	Missing data	Rows	Col-umns	Beg	End	URL
Dataset_A	0	3%	3666	8	1351-08-01	NA	https://sourceidatasetA.com
Dataset_B	2765	13%	2765	10	1351-08-01	2020-09-12	https://sourcedatasetB.com
Dataset_C	2390	19%	2390	9	1868-10-17	9999-12-31	https://sourcedatasetC.com

2) Consolidate

consolidate() allows users to produce a dataset from different datasets within the database. It resolves conflicts in data using the 'resolve' argument, which allows users to specify the value they would like to retain in the dataset.

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

Database from a package in the many packages universe

For rows and cols, select either any (all units are retained), or every (only those observations appearing in all parent datasets)

Specify value using 'resolve'

manyID	DatasetA$Beg	DatasetB$Beg	DatasetC$Beg	DatasetD$Beg
ABCEFG_1995O	1995-01-01	1995-03-04	1995-12-07	NA

`resolve = min`

manyID	Beg
ABCEFG_1995O	1995-01-01

`resolve = max`

manyID	Beg
ABCEFG_1995O	1995-12-07

`resolve = median`

manyID	Beg
ABCEFG_1995O	1995-03-04

`resolve = mean`

manyID	Beg
ABCEFG_1995O	1995-05-15

`resolve = coalesce`

manyID	Beg
ABCEFG_1995O	1995-01-01

`resolve = random`

manyID	Beg
ABCEFG_1995O	1995-03-04

if rows and cols = any

if rows and cols = every

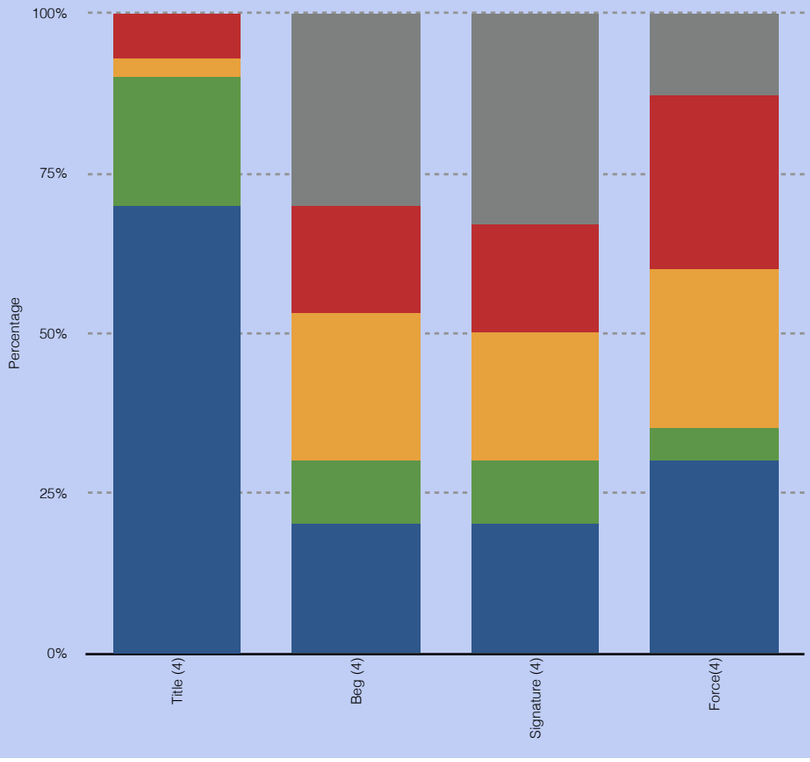
if rows = any & cols = every

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

3) Compare

```
db_plot(database, dataset, key, variable, category)
db_comp(database, dataset, key, variable, category)
```

db_plot() visualises the profile of variables across all datasets in a database (eg. agreements).
db_comp() returns a tibble that compares the variables across all datasets in the database according to the specified category/categories.



Categories

'confirmed': same across all datasets

'majority': present in most datasets

'unique': only in one dataset

'conflict': different observations in each dataset

'missing': no observations