4 Does the Network reports have everything from the remaining reports?

Note: This notebook requires a machine with 16GB of RAM + SWAP, since it loads all reports from SciELO Analytics at once, and the performed calculations require some extra memory.

Actually, the rows in each CSV from the network reports are just the rows of the respective CSV from the collection-specific reports joined together (only the network package isn't collection-specific). Also, the network reports are the only ones with "rows of intersection" between files (the joined collection-specific stuff). Below is an empirical justification for that.

In [1]:
```
import collections, glob, os
```

In [2]:
```
import numpy as np
import pandas as pd
```

4.1 CSV Types

The files available in the ZIP packages are:

- `accesses_by_journals.csv`
- `documents_affiliations.csv`
- `documents_altmetrics.csv`
- `documents_authors.csv`
- `documents_counts.csv`
- `documents_dates.csv`
- `documents_languages.csv`
- `documents_licenses.csv`
- `journals.csv`
- `journals_kbart.csv`
- `journals_status_changes.csv`

The CSV type is the name of the file without its extension, e.g. `documents_counts`.

4.2 Loading all CSV files at once

Each package have been unzipped directories named like `tabs_spa`, where `spa` is a collection code (i.e., there's just a `tabs_` leading prefix). Let's load it all in a nested dictionary structure to have a dataframe with the CSV contents in `dfs["documents_authors"]["spa"]`.

In [3]:
```
dfs = collections.defaultdict(lambda: collections.defaultdict(dict))
for fname in glob.glob("tabs_*/*.csv"):
    dname, csvname = os.path.split(fname)
    dfs[os.path.splitext(csvname)[0]][dname[5:]] = \
        pd.read_csv(fname, dtype=str, keep_default_na=False)
```

Therefore, the `tabs_network/journals.csv` file is in:

In [4]:
```
network_journals = dfs["journals"]["network"]
```

4.3 Are the rows from all `journals.csv` in `tabs_network/journals.csv`?

Yes, and every row from `tabs_network/journals.csv` are in another `journals.csv` file. To prove that, let's join the rows from every `journals.csv` source but the one from the `network`:

In [5]:
```
all_journals = pd.concat([df for k, df in dfs["journals"].items()
                          if k != "network"])
```

This joined dataframe has the same shape/size of the network journals dataframe, and no row is duplicated in these two dataframes:

In [6]:
```
{
    "all_journals": all_journals.shape,
    "all_journals (unique)": all_journals.drop_duplicates().shape,
    "network_journals": network_journals.shape,
    "network_journals (unique)": network_journals.drop_duplicates().shape,
}
```

Out [6]:
```
{'all_journals': (1732, 98),
 'all_journals (unique)': (1732, 98),
 'network_journals': (1732, 98),
 'network_journals (unique)': (1732, 98)}
```

The column names are all the same:

In [7]:
```
np.all(network_journals.columns.sort_values() ==
       all_journals.columns.sort_values())
```

Out [7]:
```
True
```

Every row is in the intersection:

In [8]:
```
pd.merge(network_journals, all_journals).shape
```

Out [8]:
```
(1732, 98)
```

And the symmetric difference is empty:

In [9]:
```
pd.concat([network_journals, all_journals]).drop_duplicates(keep=False)
```

Out [9]:

Empty DataFrame

98 columns
extraction date
study unit
collection
ISSN SciELO
ISSN's
title at SciELO
title thematic areas
title is agricultural sciences
title is applied social sciences
title is biological sciences
...
google scholar h5 2016
google scholar h5 2015
google scholar h5 2014
google scholar h5 2013
google scholar m5 2018
google scholar m5 2017
google scholar m5 2016
google scholar m5 2015

Continued on next page

98 columns
google scholar m5 2014
google scholar m5 2013

Therefore, we can say the `journals.csv` in `tabs_network` has exactly the same rows from the remaining `journals.csv` joined together.

4.4 Does `tabs_network` have all CSV types?

Yes. Every CSV type has a `network` entry:

In [10]:
```
{k: "network" in v for k, v in dfs.items()}
```

Out [10]:
```
{'documents_languages': True,
 'accesses_by_journals': True,
 'documents_dates': True,
 'documents_counts': True,
 'documents_altmetrics': True,
 'documents_authors': True,
 'journals': True,
 'journals_kbart': True,
 'documents_affiliations': True,
 'documents_licenses': True,
 'journals_status_changes': True}
```

4.5 Comparing the `network` reports with the remaining reports for all CSV types

Let's perform on every CSV type the same verification we did on `journals`:

In [11]:
```
for csv_type, datasets in dfs.items():
    print(f"Evaluating {csv_type} ...")
    network = datasets["network"]
    network_dd = network.drop_duplicates()
    remaining = pd.concat([df for k, df in datasets.items() if k != "network"])
    remaining_dd = remaining.drop_duplicates()
    shapes = [remaining.shape, remaining_dd.shape,
              network.shape, network_dd.shape]
    if len(set(shapes)) != 1:
        print(f"  There are duplicated rows or distinct sizes on {csv_type}:")
        print(f"    {shapes}")
    if np.any(network.columns.sort_values() !=
              remaining.columns.sort_values()):
        print(f"  The columns of {csv_type} aren't the same!")
        continue
    intersection = pd.merge(network_dd, remaining_dd)
    symmetric_difference = pd.concat([network_dd, remaining_dd]) \
                             .drop_duplicates(keep=False)
    if intersection.shape != shapes[1]:
        print(f"  The intersection of {csv_type} "
              "doesn't have the same number of rows!")
    if symmetric_difference.shape[0] != 0:
        print(f"  Symmetric difference of {csv_type} isn't empty!")
```

```
Evaluating documents_languages ...
Evaluating accesses_by_journals ...
```

```
Evaluating documents_dates ...
Evaluating documents_counts ...
Evaluating documents_altmetrics ...
Evaluating documents_authors ...
  There are duplicated rows or distinct sizes on documents_authors:
    [(2872098, 26), (2844742, 26), (2872098, 26), (2844742, 26)]
Evaluating journals ...
Evaluating journals_kbart ...
Evaluating documents_affiliations ...
  There are duplicated rows or distinct sizes on documents_affiliations:
    [(1690988, 26), (1415499, 26), (1690988, 26), (1415499, 26)]
Evaluating documents_licenses ...
Evaluating journals_status_changes ...
```

There are a lot of duplications going on in both `documents_affiliations` and `documents_authors`. Apart from these, the rows are unique.

The set of [distinct] rows from every network CSV are always the [distinct] rows from the remaining CSVs joined together.

4.6 Count matching

Are the duplication counts also matching?

In [12]:
```
for csv_type in ["documents_affiliations", "documents_authors"]:
    datasets = dfs[csv_type]
    print(f"Evaluating {csv_type} ...")
    network = datasets["network"]
    network_dd = network.drop_duplicates()
    network_gs = network.groupby(network.columns.tolist()).size() \
                        .rename("duplication_count") \
                        .reset_index()
    remaining = pd.concat([df for k, df in datasets.items() if k != "network"])
    remaining_gs = remaining.groupby(remaining.columns.tolist()).size() \
                            .rename("duplication_count") \
                            .reset_index()
    shapes = [(network_dd.shape[0], network_dd.shape[1] + 1),
              network_gs.shape, remaining_gs.shape]
    if len(set(shapes)) != 1:
        print(f"  The duplicated rows don't count the same on {csv_type}:")
        print(f"    {shapes}")
    intersection = pd.merge(network_gs, remaining_gs)
    symmetric_difference = pd.concat([network_gs, remaining_gs]) \
                             .drop_duplicates(keep=False)
    if intersection.shape != shapes[0]:
        print(f"  The intersection of {csv_type} "
              "w/ a duplication_count column "
              "doesn't have the expected number of rows!")
    if symmetric_difference.shape[0] != 0:
        print(f"  Symmetric difference of {csv_type} "
              "w/ a duplication_count column isn't empty!")
```

```
Evaluating documents_affiliations ...
Evaluating documents_authors ...
```

Yes, they are! =)

4.7 Duplication in CSV files besides `network`

Does any of the CSV files, individually, have duplicates?

In [13]:
```
nrowsdf = pd.DataFrame({"filename": f"tabs_{collection}/{csv_type}.csv",
                        "csv_type": csv_type,
                        "collection": collection,
                        "total_rows": dataset.shape[0],
                        "unique_rows": dataset.drop_duplicates().shape[0],
                       } for csv_type, datasets in dfs.items()
                         for collection, dataset in datasets.items()) \
            .reindex(columns=["filename", "csv_type", "collection",
                              "total_rows", "unique_rows"])
nrowsdf[nrowsdf["total_rows"] != nrowsdf["unique_rows"]]
```

Out [13]:

	filename	csv_type	collection	total_rows	unique_rows
110	tabs_ury/documents_authors.csv	documents_authors	ury	14279	14260
111	tabs_per/documents_authors.csv	documents_authors	per	35037	34374
113	tabs_col/documents_authors.csv	documents_authors	col	170355	169618
114	tabs_sza/documents_authors.csv	documents_authors	sza	63613	63194
115	tabs_bol/documents_authors.csv	documents_authors	bol	10069	10062
116	tabs_ven/documents_authors.csv	documents_authors	ven	56059	56056
117	tabs_cri/documents_authors.csv	documents_authors	cri	22856	22765
118	tabs_cub/documents_authors.csv	documents_authors	cub	115951	115937
119	tabs_bra/documents_authors.csv	documents_authors	bra	1413752	1397115
120	tabs_mex/documents_authors.csv	documents_authors	mex	150356	150000
122	tabs_arg/documents_authors.csv	documents_authors	arg	114048	113076
123	tabs_esp/documents_authors.csv	documents_authors	esp	170066	168916
125	tabs_chl/documents_authors.csv	documents_authors	chl	197798	194644
126	tabs_network/documents_-authors.csv	documents_authors	network	2872098	2844742
127	tabs_psi/documents_authors.csv	documents_authors	psi	53616	53354
128	tabs_rve/documents_authors.csv	documents_authors	rve	83395	82757
129	tabs_spa/documents_authors.csv	documents_authors	spa	145604	144402
130	tabs_sss/documents_authors.csv	documents_authors	sss	1584	1564
131	tabs_prt/documents_authors.csv	documents_authors	prt	53264	52252
176	tabs_ury/documents_affiliations.csv	documents_affiliations	ury	8189	6267
177	tabs_per/documents_affiliations.csv	documents_affiliations	per	20382	17189
178	tabs_ecu/documents_affiliations.csv	documents_affiliations	ecu	20	18
179	tabs_col/documents_affiliations.csv	documents_affiliations	col	132502	100039

Continued on next page

	filename	csv_type	collection	total_rows	unique_rows
180	tabs_sza/documents_affiliations.csv	documents_affiliations	sza	45553	39017
181	tabs_bol/documents_affiliations.csv	documents_affiliations	bol	6491	5993
182	tabs_ven/documents_affiliations.csv	documents_affiliations	ven	31500	27619
183	tabs_cri/documents_affiliations.csv	documents_affiliations	cri	17301	13879
184	tabs_cub/documents_affiliations.csv	documents_affiliations	cub	51397	50058
185	tabs_bra/documents_affiliations.csv	documents_affiliations	bra	804928	653808
186	tabs_mex/documents_affiliations.csv	documents_affiliations	mex	97770	85514
188	tabs_arg/documents_affiliations.csv	documents_affiliations	arg	64825	60439
189	tabs_esp/documents_affiliations.csv	documents_affiliations	esp	79803	72114
190	tabs_rvt/documents_affiliations.csv	documents_affiliations	rvt	345	211
191	tabs_chl/documents_affiliations.csv	documents_affiliations	chl	112992	97975
192	tabs_network/documents_affiliations.csv	documents_affiliations	network	1690988	1415499
193	tabs_psi/documents_affiliations.csv	documents_affiliations	psi	36800	34049
194	tabs_rve/documents_affiliations.csv	documents_affiliations	rve	58789	43740
195	tabs_spa/documents_affiliations.csv	documents_affiliations	spa	90207	79197
196	tabs_sss/documents_affiliations.csv	documents_affiliations	sss	963	898
197	tabs_prt/documents_affiliations.csv	documents_affiliations	prt	30231	27475

We already knew these two Network spreadsheets had duplicates, but it's clear that they aren't the only ones.

4.8 Does any duplication happen between files (besides `network`)?

No, since the sum of the number of unique rows from each CSV file matches the number of unique rows in the `network` file:

In [14]:
```
nrowsdf[(nrowsdf["collection"] == "network") &
        nrowsdf["csv_type"].isin(["documents_affiliations",
                                  "documents_authors"])]
```

Out [14]:

	filename	csv_type	collection	total_rows	unique_rows
126	tabs_network/documents_authors.csv	documents_authors	network	2872098	2844742
192	tabs_network/documents_affiliations.csv	documents_affiliations	network	1690988	1415499

In [15]:
```
nrowsdf[(nrowsdf["collection"] != "network") &
        nrowsdf["csv_type"].isin(["documents_affiliations",
                                  "documents_authors"])] \
    .groupby("csv_type").sum()
```

Out [15]:

csv_type	total_rows	unique_rows
documents_affiliations	1690988	1415499
documents_authors	2872098	2844742

4.9 Collection in `documents_affiliations` and `documents_authors`

There's a column named `collection` in both these CSV types.

The `tabs_network/documents_affiliations.csv` and `tabs_network/documents_authors.csv` have several collections.

In [16]:
```
network_coll = pd.concat([
    dfs["documents_affiliations"]["network"]
        .groupby("collection")
        .size()
        .rename("documents_affiliations.csv"),
    dfs["documents_authors"]["network"]
        .groupby("collection")
        .size()
        .rename("documents_authors.csv"),
], axis=1).sort_index()
network_coll
```

Out [16]:

collection	documents_affiliations.csv	documents_authors.csv
arg	64825	114048
bol	6491	10069
chl	112992	197798
col	132502	170355
cri	17301	22856
cub	51397	115951
ecu	20	45
esp	79803	170066
mex	97770	150356
per	20382	35037
prt	30231	53264
psi	36800	53616
rve	58789	83395
rvt	345	351
scl	804928	1413752
spa	90207	145604
sss	963	1584
sza	45553	63613
ury	8189	14279
ven	31500	56059

However, there's at most a single collection in the remaining reports. Actually, we should call each remaining report as a *collection-specific* report:

In [17]:
```
doc_coll_dict_sized = pd.merge(*[
    pd.DataFrame([
        {"collection": collection,
         csv_type + ".csv": dataset.groupby("collection").size().to_dict()}
```

```
        for collection, dataset in dfs[csv_type].items()
        if collection != "network"
    ])
    for csv_type in ["documents_affiliations", "documents_authors"]
]).set_index("collection").sort_index()
doc_coll_dict_sized
```

Out [17]:

collection	documents_affiliations.csv	documents_authors.csv
arg	{'arg': 64825}	{'arg': 114048}
bol	{'bol': 6491}	{'bol': 10069}
bra	{'scl': 804928}	{'scl': 1413752}
chl	{'chl': 112992}	{'chl': 197798}
col	{'col': 132502}	{'col': 170355}
cri	{'cri': 17301}	{'cri': 22856}
cub	{'cub': 51397}	{'cub': 115951}
ecu	{'ecu': 20}	{'ecu': 45}
esp	{'esp': 79803}	{'esp': 170066}
mex	{'mex': 97770}	{'mex': 150356}
per	{'per': 20382}	{'per': 35037}
prt	{'prt': 30231}	{'prt': 53264}
pry	{}	{}
psi	{'psi': 36800}	{'psi': 53616}
rve	{'rve': 58789}	{'rve': 83395}
rvt	{'rvt': 345}	{'rvt': 351}
spa	{'spa': 90207}	{'spa': 145604}
sss	{'sss': 963}	{'sss': 1584}
sza	{'sza': 45553}	{'sza': 63613}
ury	{'ury': 8189}	{'ury': 14279}
ven	{'ven': 31500}	{'ven': 56059}

The only collection identifier different from the reports filename suffix is `scl` for the `tabs_bra.zip` (Brazil), named differently due to its history of being the first collection. Getting the values from the collection-specific reports:

In [18]:
```
doc_coll = doc_coll_dict_sized.rename({"bra": "scl"}).drop("pry") \
    .T.apply(lambda row: row.apply(lambda cell: cell[row.name])).T \
    .sort_index()
doc_coll
```

Out [18]:

collection	documents_affiliations.csv	documents_authors.csv
arg	64825	114048
bol	6491	10069
chl	112992	197798
col	132502	170355
cri	17301	22856
cub	51397	115951
ecu	20	45
esp	79803	170066
mex	97770	150356
per	20382	35037
prt	30231	53264

Continued on next page

collection	documents_affiliations.csv	documents_authors.csv
psi	36800	53616
rve	58789	83395
rvt	345	351
scl	804928	1413752
spa	90207	145604
sss	963	1584
sza	45553	63613
ury	8189	14279
ven	31500	56059

And, as expected, that's the same in the network reports:

In [19]:
```
(doc_coll == network_coll).all()
```

Out [19]:
```
documents_affiliations.csv    True
documents_authors.csv         True
dtype: bool
```