Analysis for GARD ingest

- Spreadsheet

In [8]:
```
%reload_ext autoreload
%autoreload 2
from pathlib import Path
import os
import sys

sys.path.insert(0, Path(os.getcwd()).parent)
from gard_owl_ingest.mondo_mapping_status import gard_mondo_mapping_status

proxy_df, sssom_curate_df = gard_mondo_mapping_status(return_type='analysis')
proxy_df_exact, sssom_curate_df_exact = gard_mondo_mapping_status(
    return_type='analysis', mondo_predicate_filter=['skos:exactMatch'], gard_p
```

2023-06-29

Created set of output artefacts that have `-exact` in the filename, e.g. `gard-mondo-exact.sssom.tsv`. For these artefacts, we have filtered out everything but `skos:exactMatch` for both the Mondo::proxy and GARD::proxy mappings.

1. `gard-mondo-exact_curation.sssom.tsv` : no duplicates

FYI: `sssom_curate_df` is `gard-mondo_curation.sssom.tsv`, which is just `gard-mondo.sssom.tsv` but with additional columns.

Duplicate edges

In [24]:
```
dups = sssom_curate_df_exact[sssom_curate_df_exact.duplicated(subset=['subjec
len(dups)
```

Out[24]:0

Duplicate Mondo IDs

In [40]:
```
dup_ids = sssom_curate_df_exact[sssom_curate_df_exact.duplicated(subset=['sub
len(dup_ids)
```

Out[40]:0

2. Row difference when only using `skos:exactMatch`

There are 12,004 GARD terms. This shows that now only 10 are unmapped. This corrorborates with `gard_unmapped_terms-exact.txt`, though `gard_unmapped_terms.txt` has only 3 entries.

In [42]:
```
len(sssom_curate_df)
```

Out[42]:22378

In [43]:
```
len(sssom_curate_df_exact)
```

Out[43]:11994

2023-06-10

1. Why was unmapped_proxy_or_direct ~178 when it should have been just 3?

FYI: `proxy_df` is a subset of `gard-mondo.sssom-like.tsv`

I had a bug. I was saying that a GARD term was not mapped to Mondo if there was *any* proxy path that was unmapped between *GARD -> proxy ontology -> Mondo*. But I should have only counted as unmapped if there were *no proxy paths*. In this example below, we can see that for `GARD:15370` , there was a proxy path *`GARD:15370 -> Orphanet:888 -> MONDO:0019508`* , so it should have been counted as mapped. But since there was no proxy path to Mondo for *`GARD:15370 -> OMIM:604547`* , I had been erroneously counting as `unmapped_proxy_or_direct` in spreadsheet.

In [2]:
```
example = proxy_df[proxy_df['subject_id'] == 'GARD:15370']
example
```

Out[2]:

	subject_id	predicate_id	object_id	mondo_id	mondo_label	mondc
7381	GARD:15370	skos:broadMatch	Orphanet:888	MONDO:0019508	van der Woude syndrome	sk
7380	GARD:15370	skos:exactMatch	OMIM:604547			

2. `gard-mondo_curation.sssom.tsv` duplicates analysis

FYI: `sssom_curate_df` is `gard-mondo_curation.sssom.tsv` , which is just `gard-mondo.sssom.tsv` but with additional columns.

Duplicate subj, obj edges: Multiple possible predicates

This is where Joe needs help. Can guess what to do, but not confident enough to choose some kinds of predicates over others, either in general or case by case.

edit 2023/06/29: I think Nico helped here by determining that currently we're going to only consider exact matches: https://github.com/monarch-initiative/gard/pull/10#issuecomment-1588881615

In [36]:
```
dups = sssom_curate_df[sssom_curate_df.duplicated(subset=['subject_id', 'obje
len(dups)
```

Out[36]:6789

In [4]:
```
dups.head()
```

Out[4]:

	subject_id	predicate_id	object_id	object_label	proxy_id	r
15075	GARD:1	skos:exactMatch	MONDO:0011308	GRACILE syndrome	Orphanet:53693	
15076	GARD:1	skos:narrowMatch	MONDO:0011308	GRACILE syndrome	OMIM:603358	
7782	GARD:10000	skos:exactMatch	MONDO:0011801	spinocerebellar ataxia, autosomal recessive, w...	Orphanet:94124	
7783	GARD:10000	skos:narrowMatch	MONDO:0011801	spinocerebellar ataxia, autosomal recessive, w...	OMIM:607250	
7622	GARD:10001	skos:exactMatch	MONDO:0008964	congenital secretory chloride diarrhea 1	Orphanet:53689	

Duplicate subj, pred, obj

These exist because these are multiple GARD::Mondo mappings derived from multiple proxy terms.

In [31]:
```
dups_preds = sssom_curate_df[sssom_curate_df.duplicated(subset=['subject_id',
len(dups_preds)
```

Out[31]:20

In [33]:
```
dups_preds
```

Out[33]:

	subject_id	predicate_id	object_id	object_label	proxy_
14744	GARD:2515	skos:narrowMatch	MONDO:0009288	glycogen storage disease Ib	OMIM:23222
14743	GARD:2515	skos:narrowMatch	MONDO:0009288	glycogen storage disease Ib	OMIM:23224
20717	GARD:5184	skos:narrowMatch	MONDO:0008551	thoracolaryngopelvic dysplasia	OMIM:18776
20716	GARD:5184	skos:narrowMatch	MONDO:0008551	thoracolaryngopelvic dysplasia	OMIM:18777
4053	GARD:7183	skos:narrowMatch	MONDO:0009738	sialidosis type 2	OMIM:25615
4052	GARD:7183	skos:narrowMatch	MONDO:0009738	sialidosis type 2	OMIM:25655
12953	GARD:16523	skos:narrowMatch	MONDO:0009288	glycogen storage disease Ib	OMIM:23222
12951	GARD:16523	skos:narrowMatch	MONDO:0009288	glycogen storage disease Ib	OMIM:23224
6940	GARD:16534	skos:narrowMatch	MONDO:0001046	imperforate anus	OMIM:20750
6941	GARD:16534	skos:narrowMatch	MONDO:0001046	imperforate anus	OMIM:30180
17796	GARD:18642	skos:narrowMatch			OMIM:14174
17798	GARD:18642	skos:narrowMatch			OMIM:14233
17800	GARD:18642	skos:narrowMatch			OMIM:14247
17797	GARD:18642	skos:narrowMatch			OMIM:30543
17799	GARD:18642	skos:narrowMatch			OMIM:61356
10092	GARD:18648	skos:narrowMatch			OMIM:14174
10094	GARD:18648	skos:narrowMatch			OMIM:14233
10096	GARD:18648	skos:narrowMatch			OMIM:14247
10093	GARD:18648	skos:narrowMatch			OMIM:30543
10095	GARD:18648	skos:narrowMatch			OMIM:61356

In [35]:
```
# Showing the same thing, but leaving only the first of the duplicate rows.
# - Why do the ones with no object_id appear multiple times?
dups_preds2 = sssom_curate_df[sssom_curate_df.duplicated(subset=['subject_id'
dups_preds2
```

Out[35]:

	subject_id	predicate_id	object_id	object_label	proxy_
14743	GARD:2515	skos:narrowMatch	MONDO:0009288	glycogen storage disease Ib	OMIM:2322
20716	GARD:5184	skos:narrowMatch	MONDO:0008551	thoracolaryngopelvic dysplasia	OMIM:1877
4052	GARD:7183	skos:narrowMatch	MONDO:0009738	sialidosis type 2	OMIM:2565
12951	GARD:16523	skos:narrowMatch	MONDO:0009288	glycogen storage disease Ib	OMIM:2322
6941	GARD:16534	skos:narrowMatch	MONDO:0001046	imperforate anus	OMIM:3018
17798	GARD:18642	skos:narrowMatch			OMIM:1423
17800	GARD:18642	skos:narrowMatch			OMIM:1424
17797	GARD:18642	skos:narrowMatch			OMIM:3054
17799	GARD:18642	skos:narrowMatch			OMIM:6135
10094	GARD:18648	skos:narrowMatch			OMIM:1423
10096	GARD:18648	skos:narrowMatch			OMIM:1424
10093	GARD:18648	skos:narrowMatch			OMIM:3054
10095	GARD:18648	skos:narrowMatch			OMIM:6135

Duplicate subj, skos:exactMatch, obj

None exist!

In [7]:
```
dups_preds_exacts = dups_preds[dups_preds['predicate_id'] == 'skos:exactMatch'
len(dups_preds_exacts)
```

Out[7]:0

3. Algorithm for determining mapping predicate

A single Mondo term first can get multiple mappings to a single GARD term via multiple proxies. We can either (a) manually curate the list to get 1:1 mappings, or (b) do this algorithmically.

I set up some pseudocode for a more complex approach:

```
if preds == {'skos:narrowMatch', 'skos:exactMatch', 'skos:broadMatch'}:
    pass
elif preds == {'skos:narrowMatch', 'skos:broadMatch'}:
    pass
elif preds == {'skos:narrowMatch', 'skos:exactMatch'}:
    pass
elif preds == {'skos:exactMatch', 'skos:broadMatch'}:
    pass
```

But currently we are using this approach:

```
pred = 'skos:exactMatch' if 'skos:exactMatch' in preds \
    else 'skos:narrowMatch' if 'skos:narrowMatch' in preds \
    else 'skos:broadMatch' if 'skos:broadMatch' in preds \
    else 'skos:relatedMatch'
```

Note that as of 2023/06/29, this is moot for cases where the output artefact has `-exact` in the filename, e.g. `gard-mondo-exact.sssom.tsv` , as for these artefacts we have filtered out everything but `skos:exactMatch` for both the Mondo::proxy and GARD::proxy mappings.