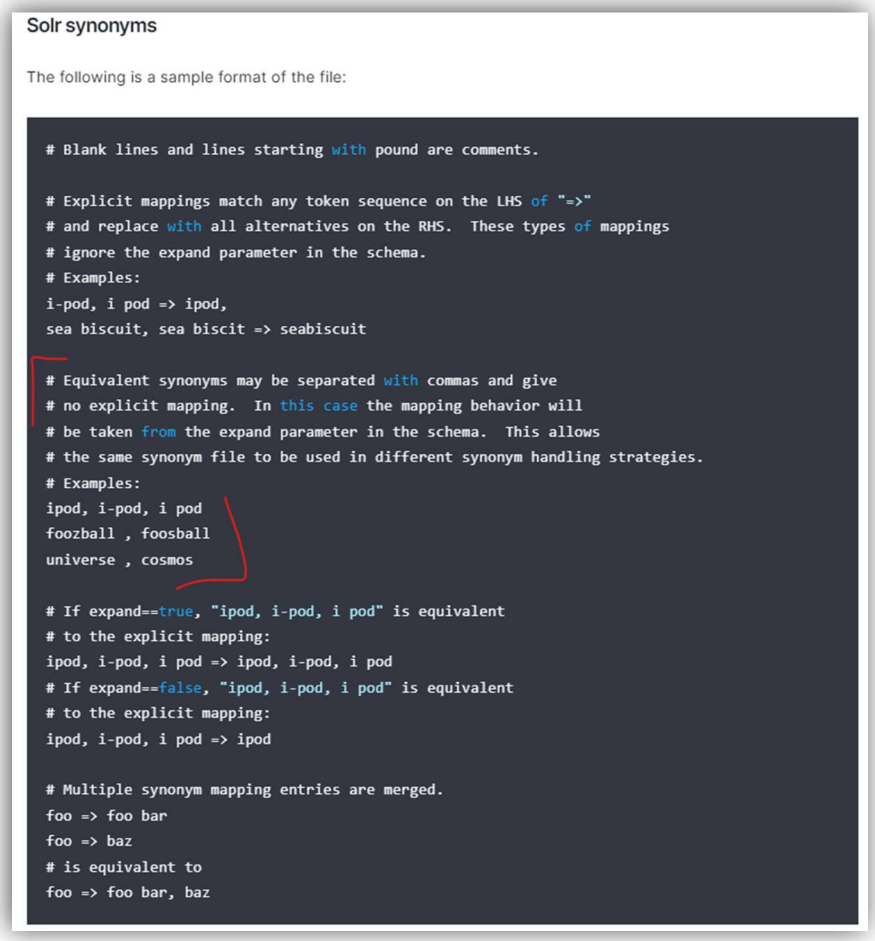
The Synonym Filter

Is it possible to implement this in Elasticsearch with Haystack? Maybe…

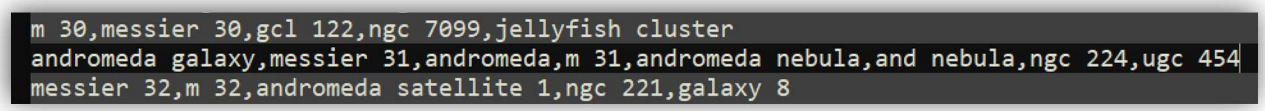
NOTES

- First you will need a synonym file (provided here as ‘astrononimcal_synonyms.csv’)

- There are multiple ways to format the synonym file, for our purposes we only needed to line up “equivalent” synonyms:
  https://www.elastic.co/guide/en/elasticsearch/reference/2.4/analysis-synonym-tokenfilter.html

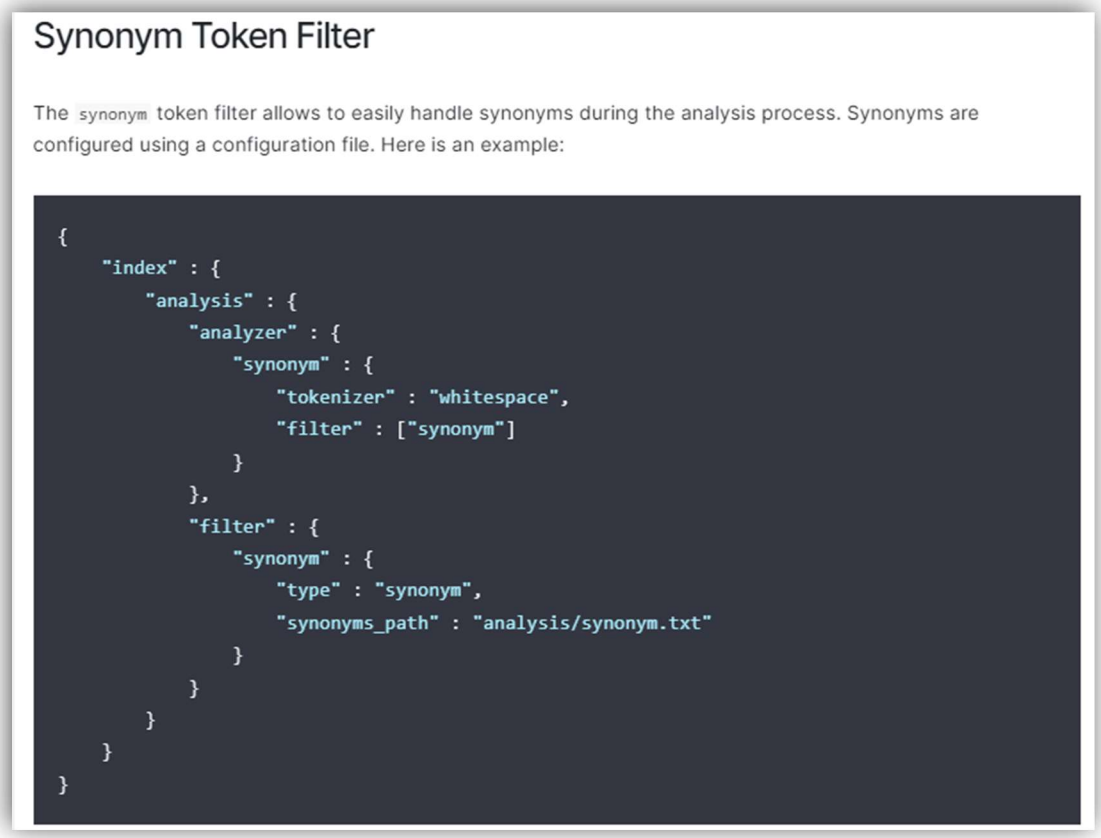


… or in our case:



- Once you have your synonym filter, the Elasticsearch `config` file will need to be updated

https://www.elastic.co/guide/en/elasticsearch/reference/2.4/analysis-synonym-tokenfilter.html



Synonym Token Filter

The `synonym` token filter allows to easily handle synonyms during the analysis process. Synonyms are configured using a configuration file. Here is an example:

```
{
    "index" : {
        "analysis" : {
            "analyzer" : {
                "synonym" : {
                    "tokenizer" : "whitespace",
                    "filter" : ["synonym"]
                }
            },
            "filter" : {
                "synonym" : {
                    "type" : "synonym",
                    "synonyms_path" : "analysis/synonym.txt"
                }
            }
        }
    }
}
```

… where "analysis/synonym.txt" is replaced with our “astrononimcal_synonyms.csv”

- Unfortunately, Haystack does not appear to allow easy changes to the Elasticsearch configurations. So, an additional tool like elasticstack will likely be needed to implement the synonym file.
  https://stackoverflow.com/questions/29254643/add-elasticsearch-synonyms-with-django-haystack
  https://github.com/bennylope/elasticstack/

- In addition to adding your synonym file to the `synonyms_path` in the Elasticsearch config file, you will notice that the synonym file I provided is standardized on all lowercase. Easily done to inputs (so they match the synonyms), *if* one can change the config file… See another config file example below.

- Additionally, you may have noticed I did not expand on catalog ID convention variations like “m 31”, “m-31”, and “m31”. Instead, I standardized all names like I do here in this following example:

```
original test string: "   #][!,@ ^&*NGc224-.99+9abc. ...   "
cleaned test string: "ngc 224 99 9 abc"

original strings: [“m 31”, “m-31”, “m31”]
cleaned strings: [“m 31”, “m 31”, “m 31”]
```

- If you need to account for all variations of searched ids such as “m 31”, “m-31”, and “m31” to be synonymous, then appropriate word delimiter token filters should also be set in the config file (*if* there is a way to do this with haystack). Alternatively, I could redo the ‘astronomical_synonyms’ to include/expand on all id convention variations instead of setting word delimiter token filters. Food for thought… here are some notes on word delimiter token filters:

https://www.elastic.co/guide/en/elasticsearch/reference/2.4/analysis-word-delimiter-tokenfilter.html

Parameters include:

`generate_word_parts`
If `true` causes parts of words to be generated: "PowerShot" ⇒ "Power" "Shot". Defaults to `true`.

`generate_number_parts`
If `true` causes number subwords to be generated: "500-42" ⇒ "500" "42". Defaults to `true`.

`catenate_words`
If `true` causes maximum runs of word parts to be catenated: "wi-fi" ⇒ "wifi". Defaults to `false`.

`catenate_numbers`
If `true` causes maximum runs of number parts to be catenated: "500-42" ⇒ "50042". Defaults to `false`.

`catenate_all`
If `true` causes all subword parts to be catenated: "wi-fi-4000" ⇒ "wifi4000". Defaults to `false`.

`split_on_case_change`
If `true` causes "PowerShot" to be two tokens; ("Power-Shot" remains two parts regards). Defaults to `true`.

`preserve_original`
If `true` includes original words in subwords: "500-42" ⇒ "500-42" "500" "42". Defaults to `false`.

`split_on_numerics`
If `true` causes "j2se" to be three tokens; "j" "2" "se". Defaults to `true`.

`stem_english_possessive`
If `true` causes trailing "'s" to be removed for each subword: "O'Neil's" ⇒ "O", "Neil". Defaults to `true`.

EXAMPLE of word delimiter token filter application in the Elasticsearch config file:
https://stackoverflow.com/questions/42484892/elasticsearch-and-word-delimiter-token-filter

```
PUT demo
{
  "settings": {
    "analysis": {
     "analyzer": {
      "index_analyzer_v1" : {
       "tokenizer" : "whitespace",
       "filter" : [ "word_delimeter"]
      }
     },
     "filter": {
      "ngram_filter" : {
       "type" : "nGram",
       "min_gram": 1,
     "max_gram": 10,
     "token_chars": [
      "letter",
      "digit"
     ]
      },
      "word_delimeter" : {
       "type" : "word_delimiter",
       "generate_number_parts" : true,
       "catenate_words" : true,
```

```
            "catenate_numbers": true,
            "preserve_original" : true,
            "stem_english_possessive": true
          }
...
```

- Lastly, FYI, the provided 'astronomical_synonyms' file only contains the names of objects found in the title data provided from the AstroBin site which had 2 or more associated names (aliases). An additional data structure (dubbed the Space Object Alias Map, AKA the SOAM) was used to help scrap object names and ids from the title data. If you see anything missing, or find any mistakes, it is likely because the SOAM either did not have the object name or ID, or the SOAM had the name or ID wrong. Let me know if you find anything, and I can see about getting it fixed / added.

  SOAM DETAILS
  https://github.com/JsonBravo/Capstone2023_DataWranglinginAstronomy/blob/main/classes_and_methods/soam_class.py

  https://github.com/JsonBravo/Capstone2023_DataWranglinginAstronomy/blob/main/data/soam_cleaned_bulk_export.json