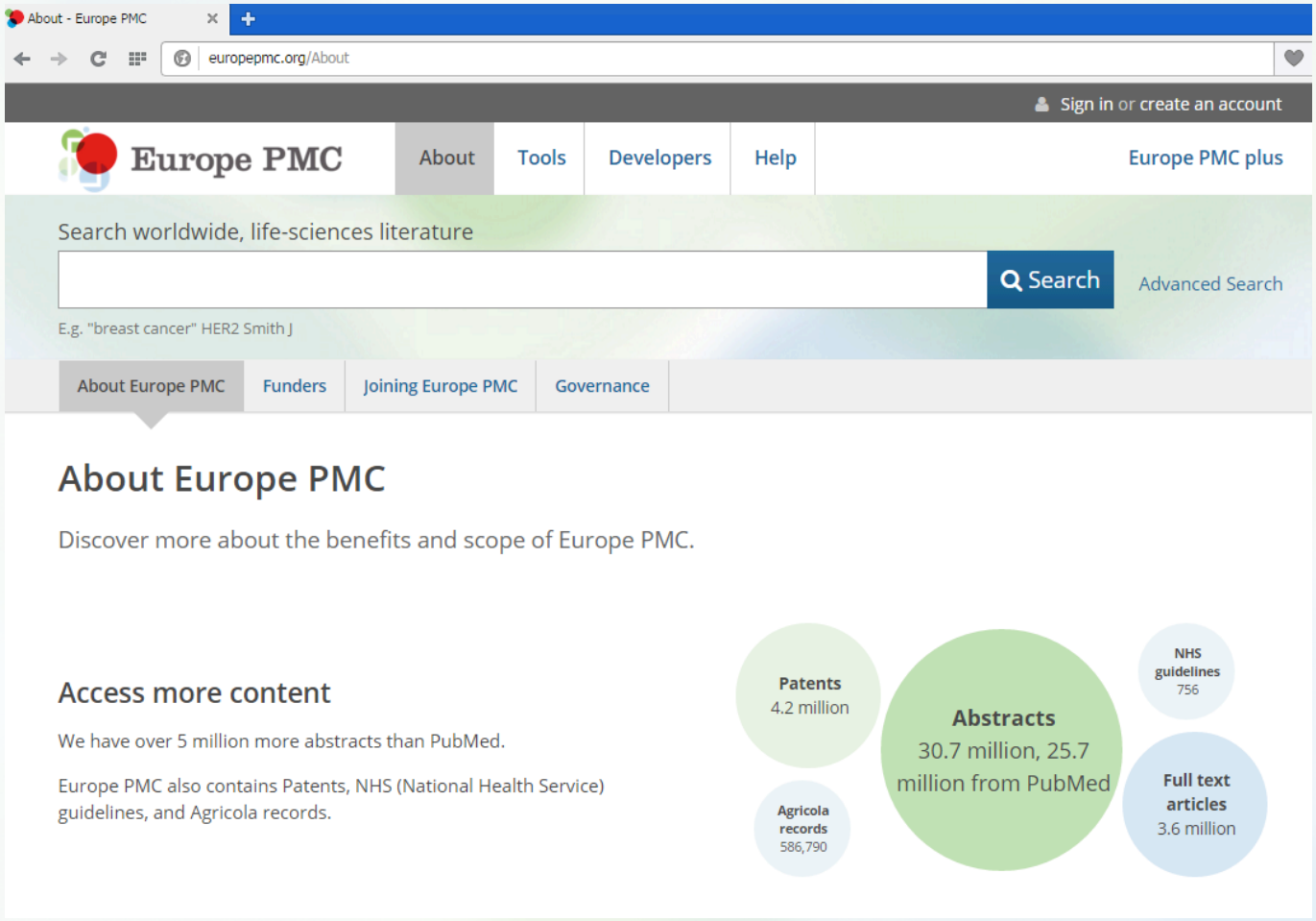
Solr

The Literature Approach

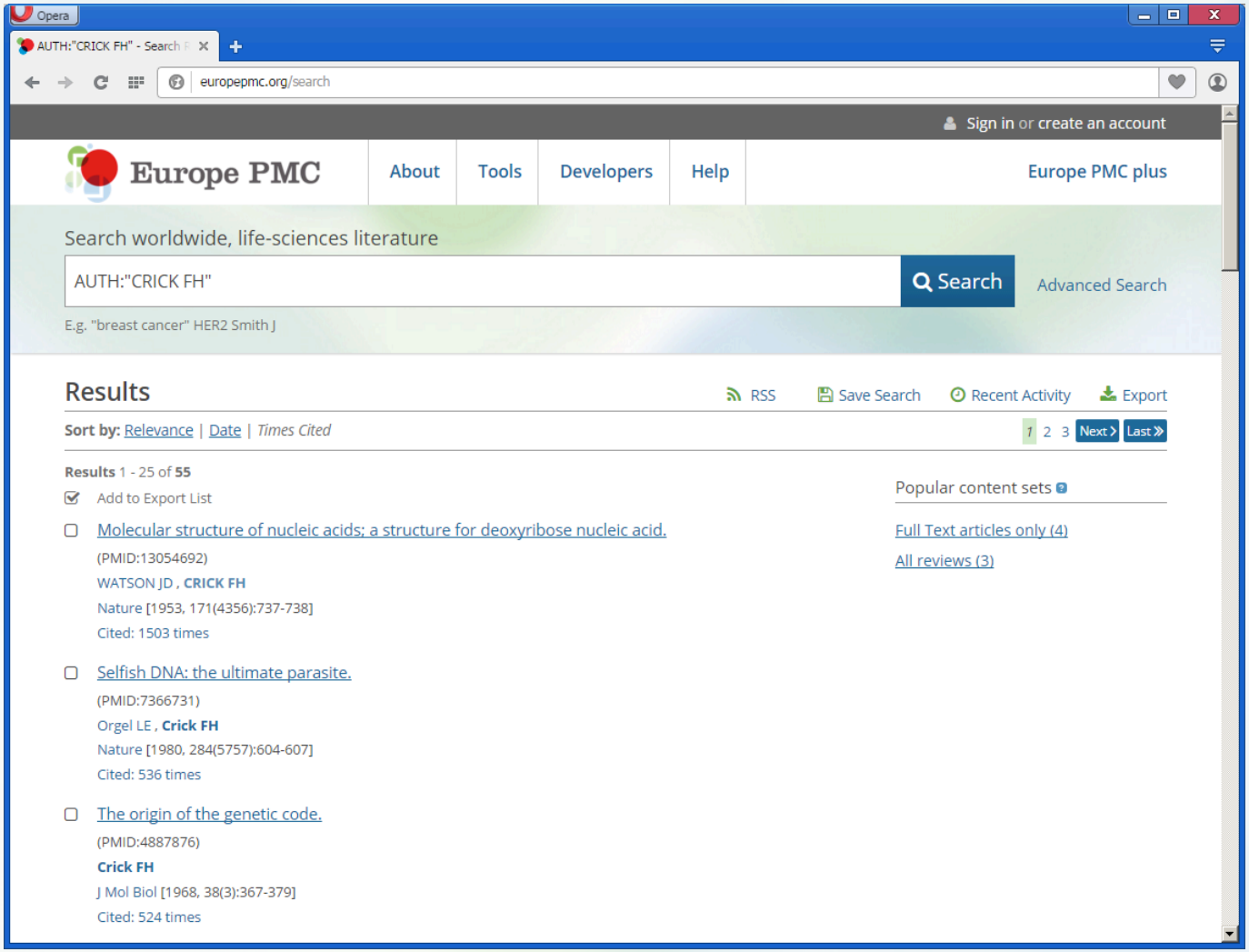
04/02/2016

Nikos Marinos
EBI - Literature Services

Literature Services – Europe PMC



Literature Services – Europe PMC



Literature Services – Europe PMC

2.1 Core bibliographic

Syntax word	Description	Example
EXT_ID:	Search for a publication by external ID: i.e. the ID assigned to a publication at repository level. Together with the publication's source, they form a unique id of the publication. Click here for more details.	EXT_ID:10826746
PMCID:	Search for a publication by its PubMed Central ID, where applicable (i.e. available as full text)	PMCID:PMC1287968
TITLE:	Search for a term or terms in publication titles	TITLE:aspirin, TITLE:"protein knowledgebase"
ABSTRACT:	Search for a term or terms in publication abstracts	ABSTRACT:malaria, ABSTRACT:"chicken pox"
PUB_YEAR:	Search by year of publication in YYYY format; note syntax for range searching.	PUB_YEAR:2000, PUB_YEAR:[2000 TO 2001]
E_PDATE:	Electronic publication date, when an article was first published online.	E_PDATE:2013-12-15, E_PDATE:20070930, E_PDATE:[2000-12-18 TO 2014-12-30], E_PDATE:[20040101 TO 20140101]
FIRST_PDATE:	The date of first publication, whichever is first, electronic or print publication. Where a date is not fully available e.g. year only, an algorithm is applied to determine the value.	FIRST_PDATE:1995-02-01, FIRST_PDATE:20000101, FIRST_PDATE:[2000-10-14 TO 2010-11-15], FIRST_PDATE:[20040101 TO 20140101]

From: Lucene

lucene-core-3.6.2.jar

Daily Job that

1. Runs One Big Oracle Query for updated data
2. Populates Java Beans
3. Builds Lucene documents
4. Synchonises index files to Live Environment
5. Reloads indexes

To: Solr

solr-core-5.2.1.jar

Daemon job that continuously

1. Runs One Big Query for updated data (Data Import Handler)

2. Sends it to the Cloud

Solr - That simple?

Java:

```
for (int i = 0; i < accessionInfo.length; i++) {
        if (!StringUtil.isNullOrEmpty(accessionInfo[i])) {
                String[] accessionData = accessionInfo[i].split(",");
                if (!tmp.contains(accessionData[0])) {
                        tmp += accessionData[0] + ",";
                        document.add(cloneFieldAndSetValue(accessionData[0].toLowerCase(), fld_ACCESSIO
                        addToFullAndFullExact(accessionData[0].toLowerCase(), document);
                }
                document.add(cloneFieldAndSetValue(accessionData[1].toLowerCase(), fld_ACCESSION_ID_2f)
                addToFullAndFullExact(accessionData[1].toLowerCase(), document);
        }
}
```

Solr:

```
<field column="ACCESSIONINFO" regex="(.*),(.*)" groupNames="ACCESSION_TYPE,ACCESSION_ID"/>

<updateRequestProcessorChain name="deduplicateMultiValued" default="true">
        <processor class="solr.UniqFieldsUpdateProcessorFactory">
                <str name="fieldRegex">ACCESSION_TYPE</str>
        </processor>
       <processor class="solr.RunUpdateProcessorFactory" />
</updateRequestProcessorChain>
```

Solr - Not always simple

- Oracle Arrays
- Oracle Structures

were resolved by custom DIH Transformers

Solr - Not always simple

```
java.sql.Struct sqlStruct = (java.sql.Struct) ob;

Object[] objStringArray = (Object[]) sqlStruct.getAttributes();

if(objStringArray.length > 0){
        aRow.put(TransformerIndexFieldConstants.HAS_FULLTEXT, objStringArray[0]);
        aRow.put(TransformerIndexFieldConstants.HAS_DOI, objStringArray[1]);
        aRow.put(TransformerIndexFieldConstants.IN_PMC, objStringArray[2]);
        String doi = (String) objStringArray[4];
        if(doi != null){
                if (doi.startsWith("http://dx.doi.org/")) {
                        doi = doi.substring(18, doi.length());
                }
        }
        aRow.put(TransformerIndexFieldConstants.DOI, doi);
        aRow.put(TransformerIndexFieldConstants.PDF, objStringArray[5]);
        aRow.put(TransformerIndexFieldConstants.FULLTEXT_SITE,oracleArrayToStringArray( objSt
}
```

```
<field column="FULLTEXT_SUMMARY" name="FULLTEXT_SUMMARY" struct="true" fields="HAS_FULLTEXT,HAS_DOI,IN_
<field column="DOI" name="DOI"  regex="http\:\/\/dx\.doi\.org\/(.*)" groupNames="DOI" />
<field column="FULLTEXT_SITE" name="FULLTEXT_SITE"  />
```

Solr - Why?

- Facets
- Dynamic sorting of results by any field
- “Did you mean”
- Auto-suggestions
- Highlighting
- Synonyms
- Author Matching
- 2-step author/journal search (e.g. “White NJ”)
- Boosting recent articles
- Close-to-Real-Time indexing
- Atomic (Partial) Updates?

Solr – Out Of the Box

- Facets (on any field)
- Dynamic sorting of results by any field
- “Did you mean” (Spell Checker): “lightly analysed” dictionary field – phrases – not ordered by hit number
- Auto-suggestions (Terms): fast – no phrases - solr.SuggestComponent "can cause EXTREMELY long startup times" (more than an hour in our case).
- Highlighting
- Boosting recent articles:

<**str** name="boost"> product(recip(ms(NOW/DAY+1YEAR, DATE_PUBLISHED), 3.16e-11,1,1),6) </**str**>

<!-- recip(x,m,a,b) : a/(m*x+b) , ms: difference in milliseconds -->

Solr – A little more fussy

Synonyms: MeSH Terms & Genes

Querying time:
- No re-indexing
- Keeps the index small
- Can easily switch synonyms on/off
- Solr doesn't handle phrase synonyms as expected
- Used a custom QueryComponent "**Better synonym handling in Solr**" by Nolan Lawson

  "move the synonym expansion from the analyzer's tokenizer chain to the query parser."
  http://nolanlawson.com/2012/10/31/better-synonym-handling-in-solr/
- Final query very complicated and very slow

Indexing time
- Faster/Simpler Queries
- Highlighting was OK??
- Larger index 600GB ➔ 770GB
- Needs reindexing (annually in our case)
- Need to have a replica non-synonyms dictionary field for switching off

synonyms.txt was too large for Zookeeper.
Had to split it in 6 parts: synonyms1.txt, synonyms2.txt,…synonyms6.txt

Solr – A little more fussy

Author Matching

"Name Search in Solr" by Doug Turnbull

```
<filter class="solr.EdgeNGramFilterFactory" minGramSize="1" maxGramSize="20" side="front"/>
```

Doug Turnbull
Turnbull, Douglas
Turnbull, Douglas G.
Turnbull, D. G.
D. Graeme Turnbull

Position	N	N+1	N+2
Standard Tokenizer:	[Douglas]	[G]	[Turnbull]
Lower Case Filter:	[douglas]	[G]	[turnbull]
NGram Filter:	[d]	[g]	[t]
	[do]		[tu]
	…(etc)…		…(etc)…
	[douglas]		[turnbull]

http://opensourceconnections.com/blog/2013/08/21/name-search-in-solr/

Solr – A little more fussy

- 2-step author/journal search: QueryComponent plugin

- Atomic updates: We need to store all the fields.

Solr – A little more fussy

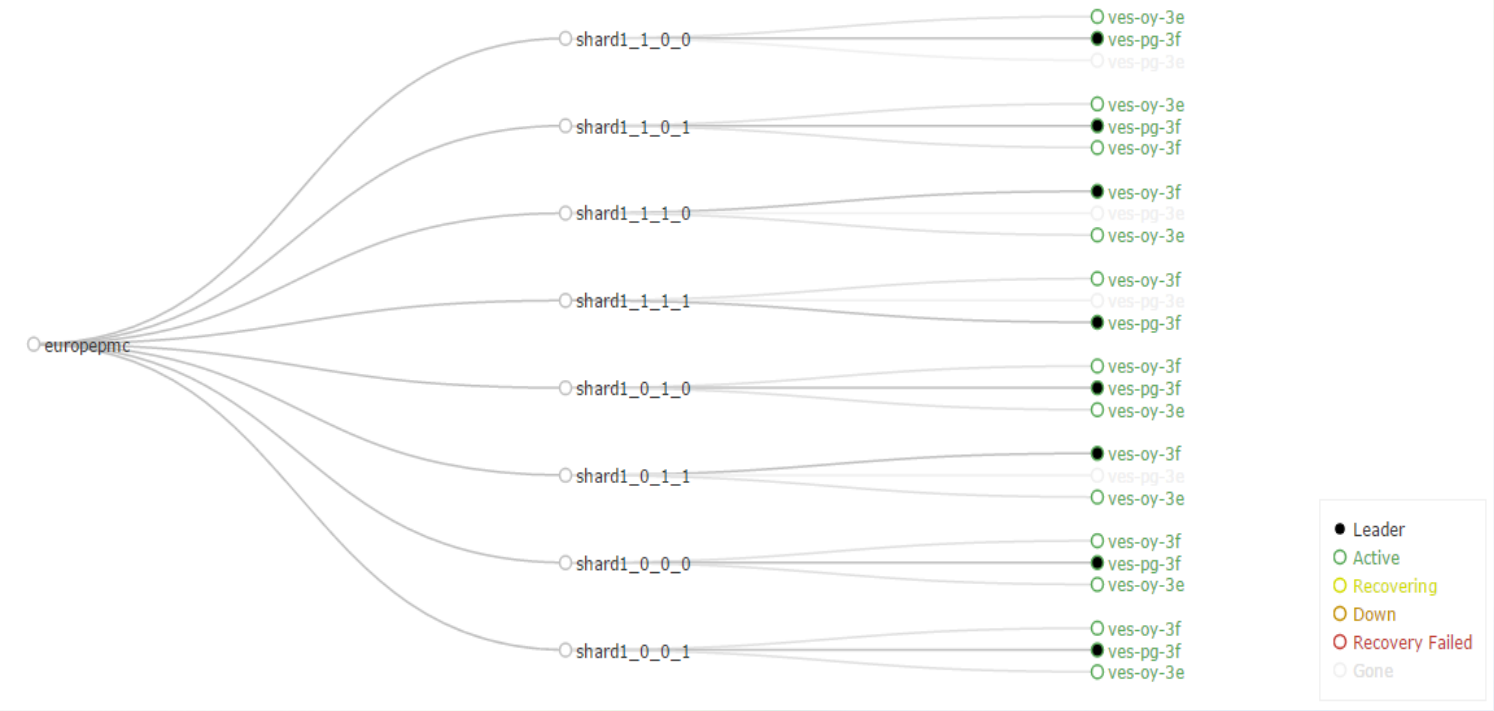
- Full Index, DIH & LSF farm: SolrJ EmbeddedSolrServer
  - 56 Jobs creating offline indexes from document.ID x to document.ID y in ~6 hours

- Merging: Overwrite default memory demanding merger.

Solr – A little more fussy

Deploy to the Cloud

1 shard-index ➔ SplitShard ➔ 2 shards ➔ SplitShard ➔ 4 shards, replication factor 3 offers efficient, redundant environment distributed over 2 VMs in each of the 2 Datacentres



Thank you

SolrQ?