서강대학교 산학협력단

Apache Solr 6.3

- 기초에서 실무까지 -

2016년 11월

아르고넷 이수명

CONTENTS

1. The basics

A. About Solr

What is Solr

- A system built to search text
  - A specialized type of database management system
- A platform to build search applications on
- Customizable, open source software

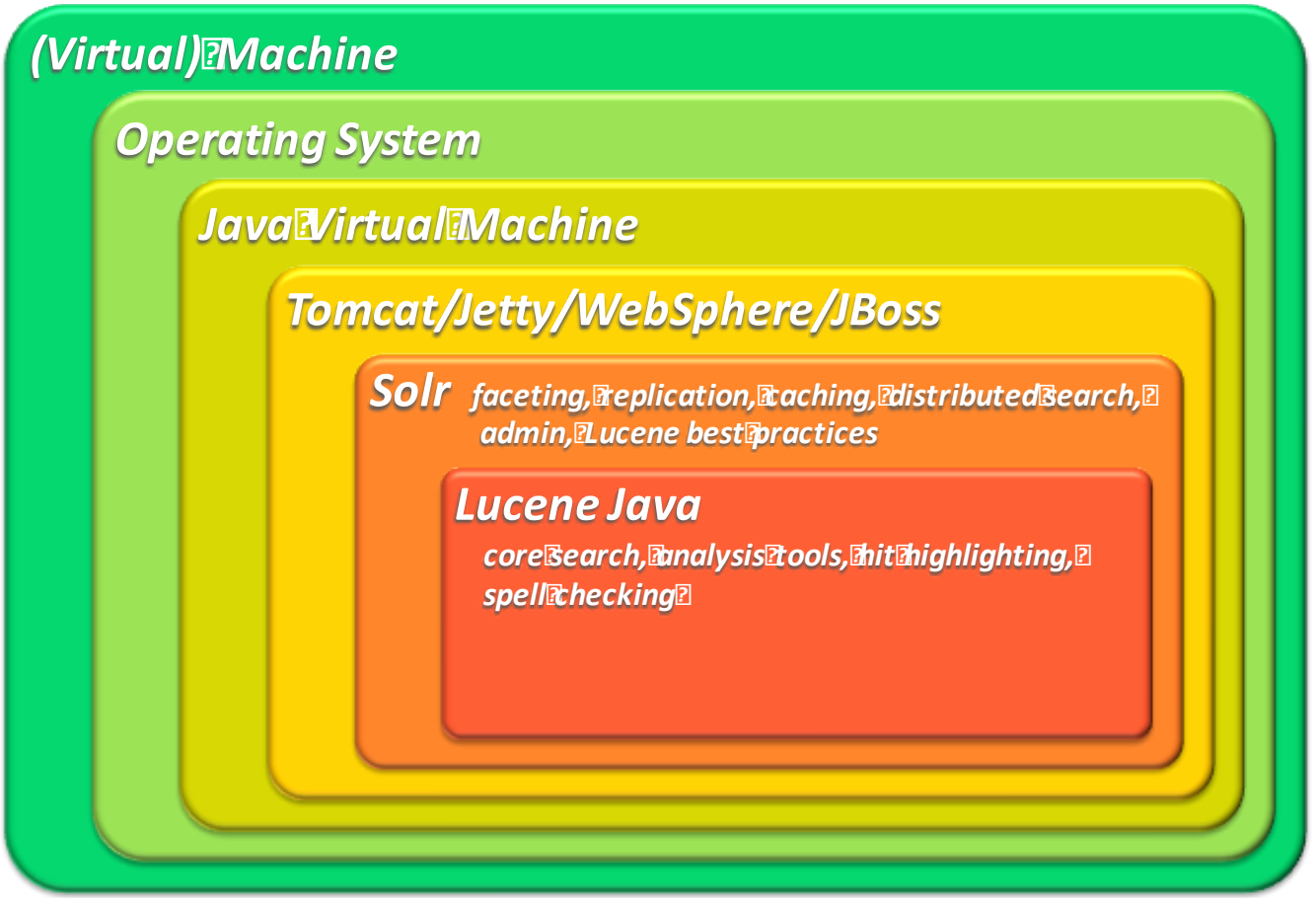
Why Solr ?

- Specialized tools do the job better
  - Solr performs much better, for text search, than a relational d atabase
  - Solr knows about languages
    - E.g. Morphological Analyzing ‘학교에’ produces ‘학교’
  - Solr has features specific to text search,
    - E.g. highlighting search results

1. The basics

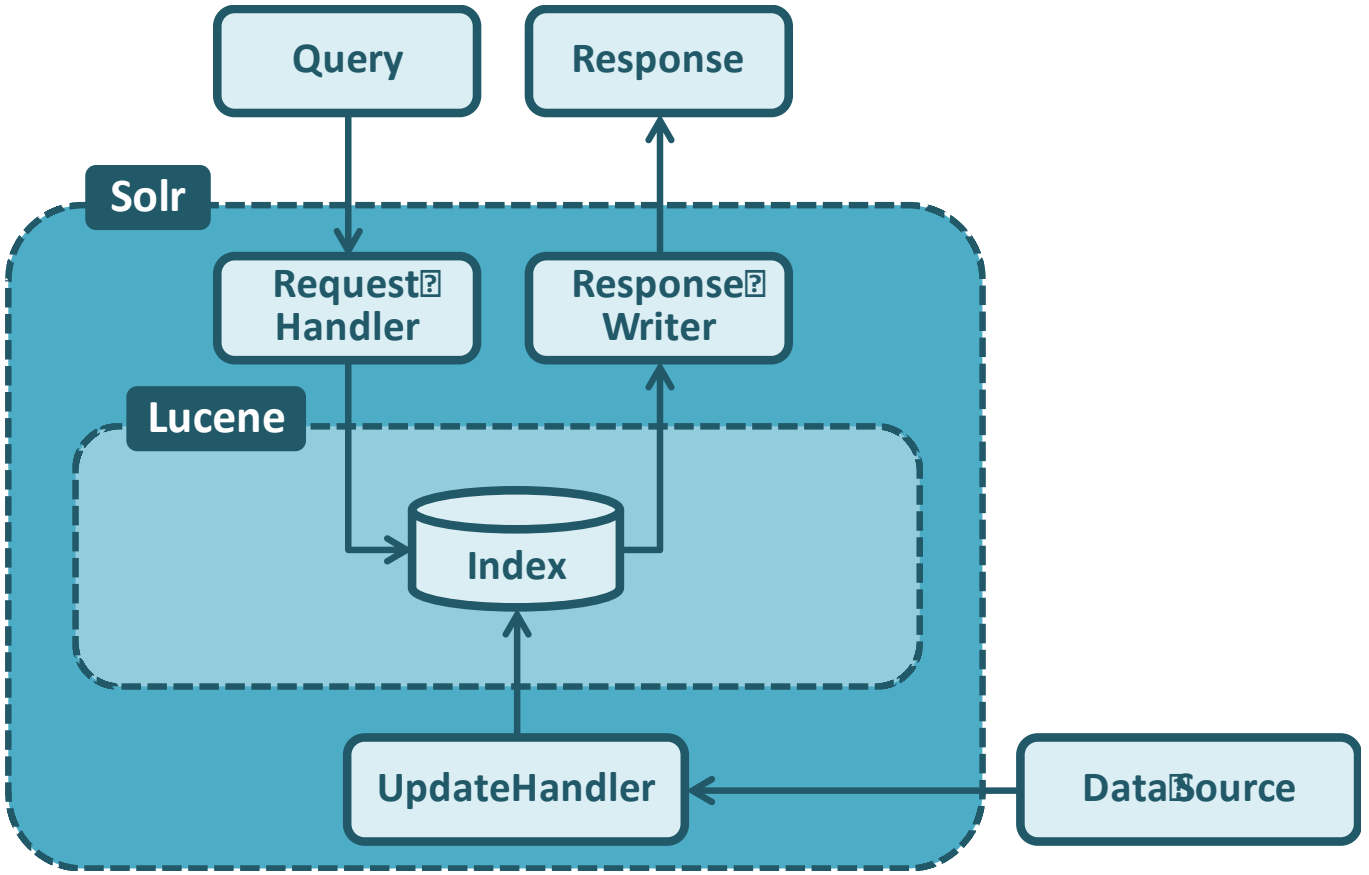
A. About Solr

Application architecture



A. About Solr

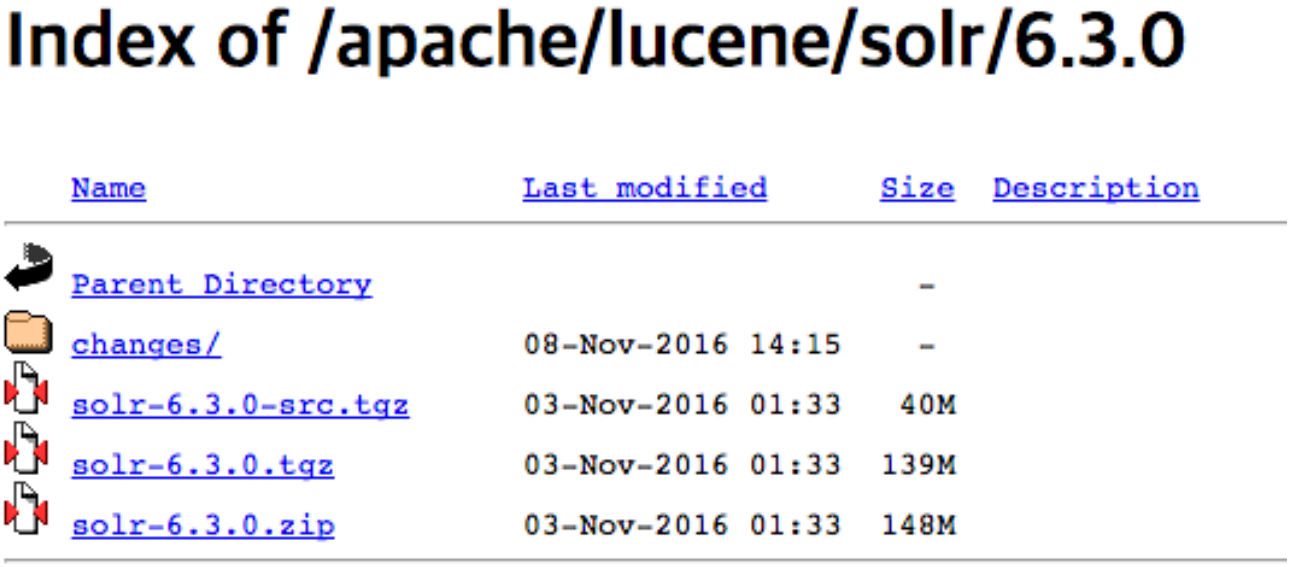
Solr block diagram



B. Installing and running Solr

❗Solr 6.0을 실행하기 위해서는 Java 1.8 이 설치되어야 함

1) Download Solr
2) Unzip Solr
3) Start Solr



Index of /apache/lucene/solr/6.3.0

Name	Last modified	Size	Description
Parent Directory		-	
changes/	08-Nov-2016 14:15	-	
solr-6.3.0-src.tgz	03-Nov-2016 01:33	40M	
solr-6.3.0.tgz	03-Nov-2016 01:33	139M	
solr-6.3.0.zip	03-Nov-2016 01:33	148M	

http://apache.tt.co.kr/lucene/solr/6.3.0/

1. The basics

C. Adding content to Solr

solr-6.1.0 — -bash — 108×37

```
isumyeong-ui-MacBook-Pro:solr-6.1.0 isumyeong$ bin/post -c techproducts example/exampledocs/*.xml
/Library/Java/JavaVirtualMachines/jdk1.8.0_45.jdk/Contents/Home/bin/java -classpath /Users/isumyeong/Documen
ts/Work/Projects/Working/Solr_Edu/20160805/solr-6.1.0/dist/solr-core-6.1.0.jar -Dauto=yes -Dc=techproducts -
Ddata=files org.apache.solr.util.SimplePostTool example/exampledocs/gb18030-example.xml example/exampledocs/
hd.xml example/exampledocs/ipod_other.xml example/exampledocs/ipod_video.xml example/exampledocs/manufacture
rs.xml example/exampledocs/mem.xml example/exampledocs/money.xml example/exampledocs/monitor.xml example/exa
mpledocs/monitor2.xml example/exampledocs/mp500.xml example/exampledocs/sd500.xml example/exampledocs/solr.x
ml example/exampledocs/utf8-example.xml example/exampledocs/vidcard.xml
SimplePostTool version 5.0.0
Posting files to [base] url http://localhost:8983/solr/techproducts/update...
Entering auto mode. File endings considered are xml,json,jsonl,csv,pdf,doc,docx,ppt,pptx,xls,xlsx,odt,odp,od
s,ott,otp,ots,rtf,htm,html,txt,log
POSTing file gb18030-example.xml (application/xml) to [base]
POSTing file hd.xml (application/xml) to [base]
POSTing file ipod_other.xml (application/xml) to [base]
POSTing file ipod_video.xml (application/xml) to [base]
POSTing file manufacturers.xml (application/xml) to [base]
POSTing file mem.xml (application/xml) to [base]
POSTing file money.xml (application/xml) to [base]
POSTing file monitor.xml (application/xml) to [base]
POSTing file monitor2.xml (application/xml) to [base]
POSTing file mp500.xml (application/xml) to [base]
POSTing file sd500.xml (application/xml) to [base]
POSTing file solr.xml (application/xml) to [base]
POSTing file utf8-example.xml (application/xml) to [base]
POSTing file vidcard.xml (application/xml) to [base]
14 files indexed.
COMMITting Solr index changes to http://localhost:8983/solr/techproducts/update...
Time spent: 0:00:00.239
isumyeong-ui-MacBook-Pro:solr-6.1.0 isumyeong$
```

D. Reading a Solr XML response

```
<response>
  <lst name="responseHeader">
    <int name="status">0</int>
    <int name="QTime">0</int>
    <lst name="params">
      <str name="q">*:*</str>
      <str name="indent">on</str>
      <str name="wt">xml</str>
    </lst>
  </lst>
  <result name="response" numFound="32" start="0">
    <doc>
      <str name="id">GB18030TEST</str>
      <str name="name">Test with some GB18030 encoded characters</str>
      <arr name="features">
        <str>No accents here</str>
        <str>这是一个功能</str>
        <str>This is a feature (translated)</str>
        <str>这份文件是很有光泽</str>
        <str>This document is very shiny (translated)</str>
      </arr>
      <float name="price">0.0</float>
      <str name="price_c">0.0,USD</str>
      <bool name="inStock">true</bool>
      <long name="_version_">1540304712393293824</long>
    </doc>
```

What you get

- Success / failure
- Time to process query
- Number of matching documents
- The first N documents
- …and more

Response writers

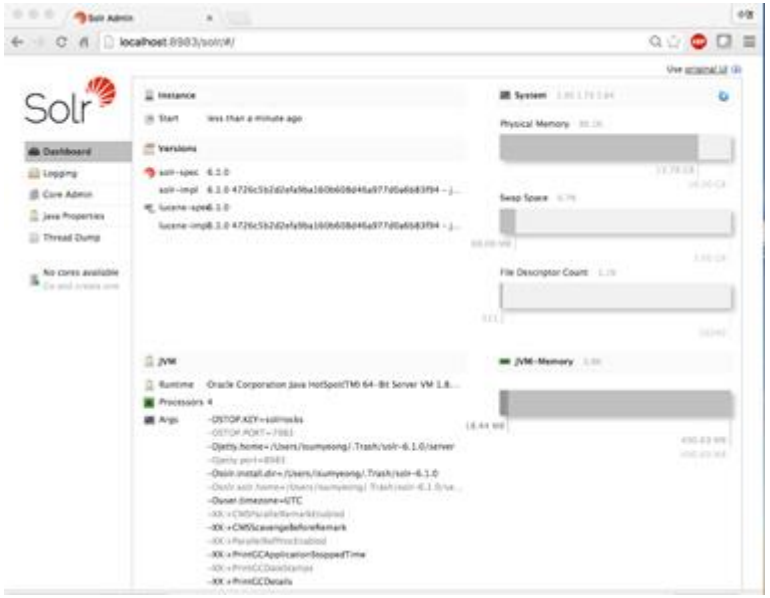
- Response writers format the output fr om requests
- Many types
  - XML
  - JSON
  - Ruby
  - PHP
  - CSV
- &wt=json

Lab 1. Getting Solr up and running

- Windows 일때 데이터 색인 명령어
  java -Dauto -Dc=techproducts -Dfiletypes=xml -jar example₩exampledocs₩post.jar example₩exampledoc₩*.xml

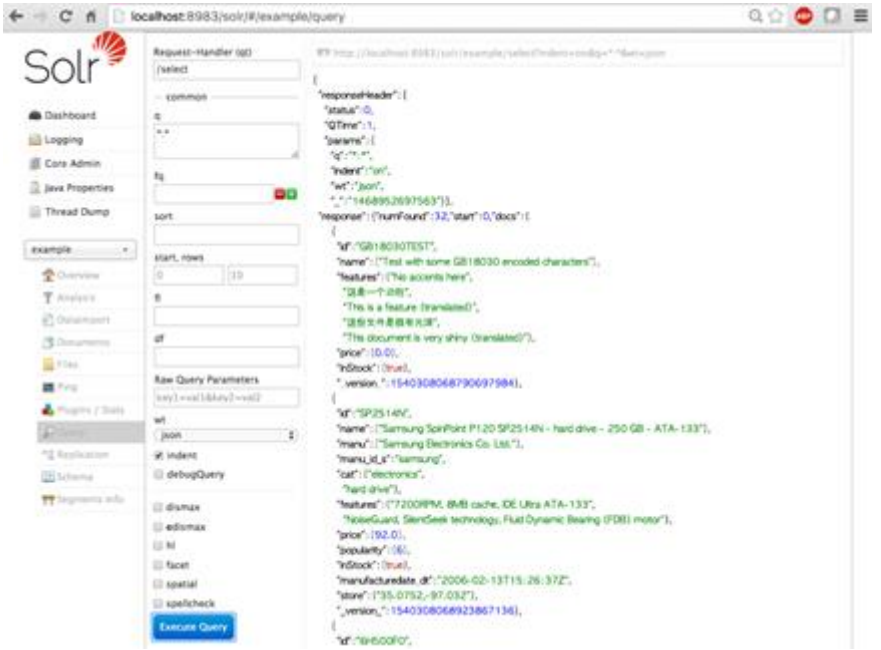
Step 1.

1. solr-6.1.0.zip 의 압축을 풀어 푼다.
2. $Solr.Home 폴더로 이동한다.
   ex) cd solr-6.1.0
3. solr 를 실행한다.
   bin/solr start
4. 브라우저에서 http://localhost:8983/solr 입력하여 Admin Tool을 확인한다.



Step 2.

1. **Core 생성** : bin/solr create -c techproducts -d server/solr/configsets/sample_techproducts_configs
2. **데이터 색인** : bin/post -c techproducts example/exampledocs/*.xml
3. Admin Tool의 core selector에서 techproducts 을 선택하고 query 메뉴를 선택
4. q필드에 "*:*" 또는 "ipod'을 입력 후 Execute Query 버튼 클릭



E. Changing parameters in the URL

http://localhost:8983/solr/techproducts/select?q=samsung

http://localhost:8983/solr/techproducts/select?q=samsung&fl=id,name

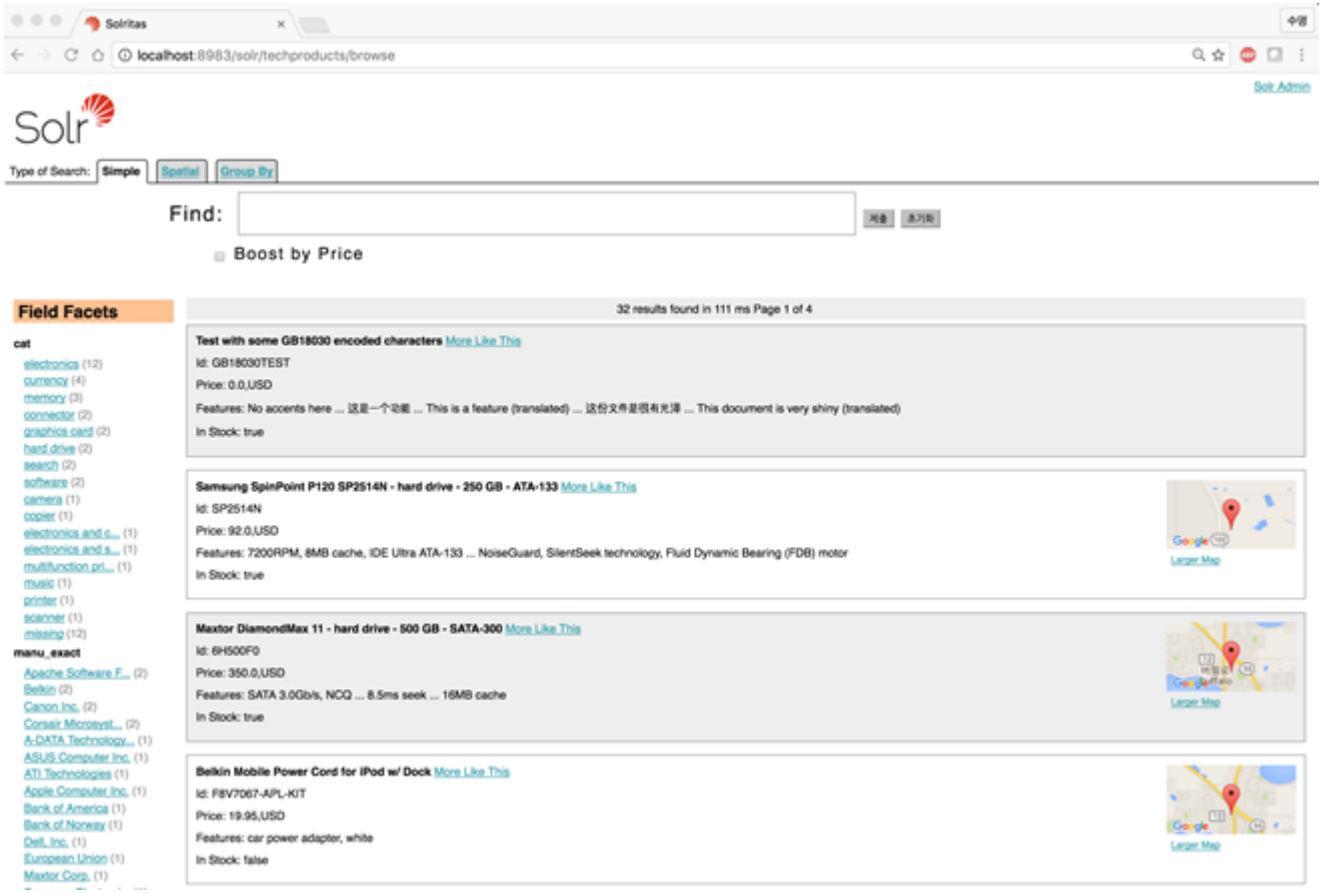
http://localhost:8983/solr/techproducts/select?q=samsung&fl=id,name&wt=json

http://localhost:8983/solr/techproducts/select?q=samsung&fl=id,name&wt=json&omitHeader=on

http://localhost:8983/solr/techproducts/select?q=samsung&fl=id,name&wt=json&indent=on

F. Using the browse interface

- UI를 만들지 않고 검색결과를 데모하는 목적으로 Browser Interface를 사용할 수 있음.
- 본격적인 기능을 개발하기 전에 POC(Proof of concept)을 위해 활용하면 좋음



F. Using the browse interface – Queries

Range queries

- 일정한 범위에 존재하는 문서를 검색하기 위한 파라미터
- 양쪽 끝의 값을 포함하는 경우는 []
  - price:[0 TO 92]
  - price:[0 TO *]
  - price:[3 TO *]
- 양쪽 끝의 값을 포함하지 않는 경우 {}
  - price:{0 TO 92}
- {와]을 함께 사용할 수도 있음
  - price:[0 TO 92}

Date queries

- Date format
  - 1995-12-31T23:59:59Z
- Date math
  - NOW
  - <fielc name="timestamp" type="date" indexed="true" stored="true" defalut="NOW"/>
  - NOW/YEAR
    2010-07-28T23:34:45Z => 2010-01-01T00:00:0Z
  - NOW/HOUR, NOW/SECOND
  - NOW/YEAR-1YEAR+2DAYS

Filter queries

- main query의 검색결과를 줄여주기 위해 사용
- ordering이나 scoring에 변화를 주지 않는다
- 사용 사례
  - &fq=cat:electronics&fq=price:[0 TO 100]&fq=rating:[3 TO *]

Lab 2. Using the browse interface

1. “bin/solr stop –all”으로 실행중인 Solr를 중지한다
2. Solr에 내장된 techproducts 예제 core로 solr를 실행한다.

   bin/solr –e techproducts
3. http://localhost:8983/solr/techproducts/browse 로 접속
4. 좌측 Field Facets 영역에서 cat 항목에서는 electronics을 선택하고 manu_exact 항목에서는 Belkin을 선택한 후 브라우저의 주소창에서 URL이 어떻게 변하는지 확인
5. price가 $179와 $330 사이에 있는 모든 아이템을 검색하고 검색결과가 몇건인지 확인
6. manufactured_dt 가 1/1/2007 이전인 모든 아이템을 검색하고 검색결과가 몇건인지 확인
7. San Francisco에서 10k 이내에 있는 모든 아이템을 검색하고 검색결과가 몇건인지 확인
8. 검색어 Dell로 검색을 하고 검색결과가 몇건인지 확인
9. 검색어 dell로도 검색을 하고 이전의 검색건수와 비교
10. d* 로 검색을 하고 검색결과가 몇건인지 확인
11. D* 로 검색을 하고 이전의 검색건수와 비교

2. Searching

A. Sorting results

- Solr는 Score, 필드의 Value, Function에 의해 Sorting할 수 있다.
- 오름차순(ascending) 또는 내림차순(descending)에 의해 Sorting할 수 있다.
- 아래와 같이 다중 필드도 가능하다.
  - &sort=id asc,manu_id_s desc

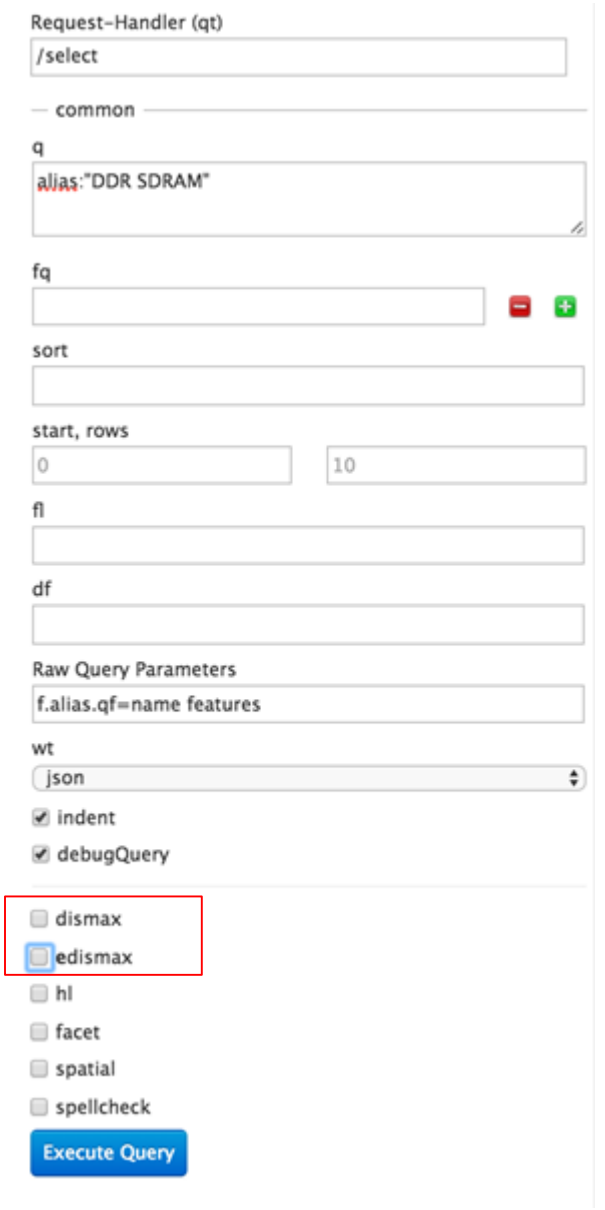
Lab 3. Sorting results

1. price 필드의 값에 대해 내림차순으로 정렬하시오.
2. q=includes:[* TO *]으로 검색을 하고 includes 필드의 값으로 내림차순, 오름차순으로 각각 정렬해 보시오
   - 결과에 변화가 있는지 확인하라. 없다면 어떻게 수정할 수 있을까?
3. q=*:* 으로 검색을 하고 price에 대해 내림차순, id에 대해 내림차순으로 정렬하시오.

2. Searching

B. Query parsers

- Standard query Parser
  - Default query parser for Solr
  - Also known as Lucene query parser
  - Robust and fairly intuitive syntax allowing
  - Malformed queries can cause errors
- Dismax query parser
  - Dismax syntax is very simple
- Edismax query parser
  - Used by the /browse interface
  - Handles errors more gracefully
  - Uses Lucene query parser by default
  - on Errors, uses dismax



2. Searching

C. More queries

- Boolean quries
  - +this -that
  - this AND that
- Field queries
  - title:this
  - description:that
- Range queries
  - price:[0 TO 100]
  - -price:[50 TO 60]
- Phrase/proximity queries
  - “Harry Potter” matches only Harry Potter
  - “HarryPotter”~1 matches Harry James Potter
- Multi-term queries
  - title:apple pie
  - title:(apple pie)
- Combine them
  - +this -title:that +price:[100 To *] -name:”Harry Potter”
- Escaping special characters
  - Query는 Solr에 전송될 때 URL Encoding이 되어야 함

Lab 4. Morequeries

1. http://locahost:8983/solr/#/techproducts/query에 접속
2. debugQuery 체크박스와 edismax 체크박스가 check 되도록 한다.
3. q 필드에 name:ipod을 입력한다.
4. 검색결과 아랫 부분으로 내려가서 “parsedquery”을 확인한다.
5. 다음의 쿼리를 각각 입력하여 검색결과를 확인한다.
    - *:*
    - price:{0 TO 100]
    - Ipod touch
6. qf 필드에 name을 입력하여 Ipod touch로 검색을 실행하고 결과를 확인한다.
7. manufacturedate_dt 가 2005년인 것을 제외하고 검색한다.
8. ipod은 포함되나 apple은 제외하고 검색한다.
9. “timing unbuffered” 구(Phrase)에 대해 검색을 한다. 검색결과가 존재하는가?
10. 존재하지 않으면, 단어들 간의 간격이 얼마나 떨어지도록 조건을 주어야 검색되는가?

    Hint) Phrase/proximity queries 참조

2. Searching

D. Hardwiring request parameters

- solrconfig.xml (Solr's configuration)
  - Controls request handlers, search components, and more
  - $SOLR_HOME/server/solr/techproducts/conf/solrconfig.xml

```
<requestHandler name="/browse" class="solr.SearchHandler">
   <lst name="defaults">
     <str name="echoParams">explicit</str>

     <!-- VelocityResponseWriter settings -->
     <str name="wt">velocity</str>
     <str name="v.template">browse</str>
     <str name="v.layout">layout</str>
     <str name="title">Solritas</str>

     <!-- Query settings -->
     <str name="defType">edismax</str>
     <str name="qf">
        text^0.5 features^1.0 name^1.2 sku^1.5 id^10.0 manu^1.1 cat^1
        title^10.0 description^5.0 keywords^5.0 author^2.0 resourcename^1.0
     </str>
     <str name="mm">100%</str>
     <str name="q.alt">*:*</str>
     <str name="rows">10</str>
     <str name="fl">*,score</str>

     <str name="mlt.qf">
       text^0.5 features^1.0 name^1.2 sku^1.5 id^10.0 manu^1.1 cat^1.4
       title^10.0 description^5.0 keywords^5.0 author^2.0 resourcename^1.0
     </str>
     <str name="mlt.fl">text,features,name,sku,id,manu,cat,title,description,keywords,author,resourcename</str>
     <int name="mlt.count">3</int>
```

```
<requestHandler name="/select" class="solr.SearchHandler">
  <!-- default values for query parameters can be specified, these
       will be overridden by parameters in the request
    -->
   <lst name="defaults">
     <str name="echoParams">explicit</str>
     <int name="rows">10</int>
     <!-- Controls the distribution of a query to shards other than itself.
          Consider making 'preferLocalShards' true when:
            1) maxShardsPerNode > 1
            2) Number of shards > 1
            3) CloudSolrClient or LbHttpSolrServer is used by clients.
          Without this option, every core broadcasts the distributed query to
          a replica of each shard where the replicas are chosen randomly.
          This option directs the cores to prefer cores hosted locally, thus
          preventing network delays between machines.
          This behavior also immunizes a bad/slow machine from slowing down all
          the good machines (if those good machines were querying this bad one).

          Specify this option=false for clients connecting through HttpSolrServer
     -->
```

E. Adding fields to default search

- Solr는 필드를 지정하지 않으면 디폴트 필드에 대해 검색한다.
  - example 에서는 text 필드가 디폴트 필드임
- 사용자는 다른 필드에 대해 쿼리에 의해 검색할 수 있다.
  - author:rowling
  - title:hallows
- solrconfig.xml에서 설정할 수도 있다.
- 모든 필드를 검색하는 것은 느리다.
- 여러 필드를 하나의 필드에 복사해서 검색한다.

```
<copyField source="name" dest="text"/>
<copyField source="author" dest="text"/>
<copyField source="summary" dest="text"/>
```

Lab 5. Default search fields

1. http://locahost:8983/solr/#/techproducts/query에 접속하여 shiny로 검색
2. shiny가 어느 필드에 존재하는지 확인
3. $SOLR_HOME/server/solr/techproducts/conf/managed-schema 파일을 열어서 아래의 copyField 를 확인

```
<copyField source="cat" dest="text"/>
<copyField source="name" dest="text"/>
<copyField source="manu" dest="text"/>
<copyField source="features" dest="text"/>
<copyField source="includes" dest="text"/>
<copyField source="manu" dest="manu_exact"/>
```

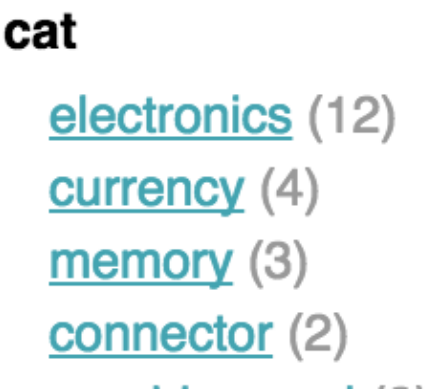
4. features가 있는 행을 제거한다.
5. solr를 재실행한다 : bin/solr restart
6. 재 색인한다 : bin/post -c techproducts example/exampledocs/*.xml
7. shiny로 검색결과가 없는 것을 확인
8. /browse에서 shiny로 검색하여 검색결과가 존재하는 것을 확인
9. solrconfig.xml 파일을 열어서 /browse request handler 설정 하위의 qf 테그를 찾는다
10. qf 테그에서 features^1.0 을 제거하고 solr 를 재실행한다.
11. /browse 에서 shiny 로 다시 검색하여 결과가 없는 것을 확인한다.

F. Faceting (개요)

- Facet의 대상
  - Field에 저장된 값
    - string, date, number 등 field type 의 값
    - Multiple type 의 값
    - separated valued는 부적절
  - Queries
    - Range queries가 가장 일반적인 유형
      (This year / Last year)
- Facet의 유형
  - Field facet
  - Query facet
  - Range facet
  - Hierarchical facet
  - Pivot facet

F. Faceting (Field/Query Facet)

- Field Facet
  - Query에 의한 방법
    - facet.field=cat
  - solfconfig.xml에 의한 방법
    - <str name="facet.field">cat</str>



- Query Facet
  - Query에 의한 방법
    - facet.query=price:[50 TO 200]
  - solfconfig.xml에 의한 방법
    - <str name="facet.query">price:[50 TO 200]</str>

F. Faceting (Range Facet)

facet.range=manufacturedate_dt

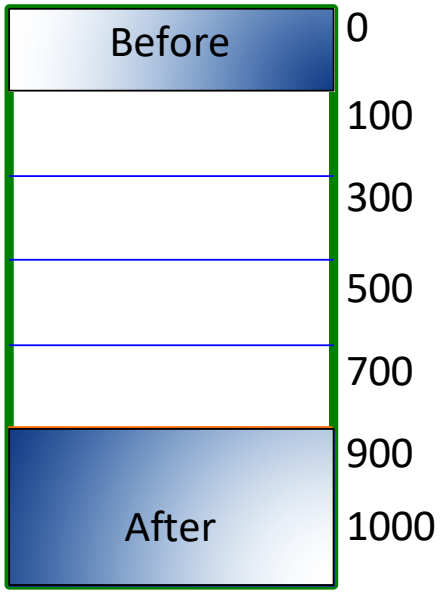
f.manufacturedate_dt.facet.range.start=NOW/YEAR-10YEARS

f.manufacturedate_dt.facet.range.end=NOW

f.manufacturedate_dt.facet.range.gap=+1YEAR

f.manufacturedate_dt.facet.range.other=before

f.manufacturedate_dt.facet.range.other=after



Range Facets

price

50.0 - 100.0 (2)

150.0 - 200.0 (2)

popularity

3 - 6 (1)

6 - 9 (3)

manufacturedate_dt

Less than 2006-01-01T00:00:00Z (0)

2006-01-01T00:00:00Z - 2006-01-01T00:00:00Z+1YEAR (3)

F. Faceting (Hierarchical facet)

Display

- **Books (2,123,456)**
  - Computers & Technology (601,234)
    - Computer Science (123,456)
      - Artificial Intelligence (27,665)
      - Human-Computer Interaction (1,353)
      - Information Theory (2,004)
      - …

Field Type

```
<fieldType name="descendent_path" class="solr.TextField">
   <analyzer type="index">
     <tokenizer class="solr.PathHierarchyTokenizerFactory" delimiter="/" />
   </analyzer>
   <analyzer type="query">
     <tokenizer class="solr.KeywordTokenizerFactory" />
   </analyzer>
</fieldType>
```

F. Faceting (Pivot facet)

- Finds the top N constraints for field1
- Then, for each of those
  - finds the top N constraints for field2, etc
- Syntax: facet.pivot=field1,field2,field3,…facet.pivot=cat,inStock

	#docs	#docs with inStock:true	#docs with instock:false
cat:electronics	14	10	4
cat:memory	3	3	0
cat:connector	2	0	2
cat:graphics card	2	0	2
cat:hard drive	2	2	0

Lab 6. Faceting

1. http://locahost:8983/solr/techproducts/browse?facet=on&facet.field=features 에 접속
2. features의 facet을 확인하고 문제를 파악한다.
3. $SOLR_HOME/server/solr/techproducts/conf/managed-schema를 텍스트 편집기로 연다.
4. features 필드를 정의한 부분을 찾는다.
5. features_exact 이란 이름으로 features 필드를 복사하고 type은 string으로 수정한다.
6. copyField 필드를 추가하여 features 필드의 값을 feature_exact 필드로 복사하도록 설정한다.
7. Solr 재시작 및 재색인 후 features_exact로 facet 을 확인한다.
8. price facet의 gap이 100이 되도록 수정한 후 확인한다.
9. manufactureddate_dt의 gap이 1달이 되도록 수정한 후 확인한다.
10. material 폴더에서 hierarchical.xml 파일을 열어서 hierarchical_category 필드 값의 구조를 확인한다.
11. managed-schema파일을 열어서 descendent_path field type 으로 hierarchical_category 필드를 추가한다.
12. Solr를 재시작 후, hierarchical.xml 파일을 색인한다.
13. http://locahost:8983/solr/techproducts/browse?facet=on&facet.field=hierarchical_category 로 facet을 확인한다.

2. Searching

G. Grouping

- Field value 에 의해 grouping
- Field가 multiple value 인 경우는 Grouping 할 수 없음
- Syntax:

  group=true&group.field=manu_exact&group.limit=2&group.sort=id asc&sort=manu_exact asc

Maxtor Corp. (1)

Maxtor DiamondMax 11 - hard drive - 500 GB - SATA-300 More Like This

Id: 6H500F0

Price: 350.0,USD

Features: SATA 3.0Gb/s, NCQ ... 8.5ms seek ... 16MB cache

In Stock: true

Larger Map

Belkin (2)

Belkin Mobile Power Cord for iPod w/ Dock More Like This

Id: F8V7067-APL-KIT

Price: 19.95,USD

Features: car power adapter, white

In Stock: false

Larger Map

A. Adding your own content to Solr

Atomic updates

- Since 4.0, Solr can update documents
- For this to work, all fields in your document must be stored, not just indexed

```
$ curl http://localhost:8983/solr/update -H 'Content-type:application/json'
-d '[
 {"id"        : "book1",
  "pubyear_i" : { "add" : 2001 },
  "ISBN_s"    : { "add" : "0-380-97365-1"}
 }
]'
$ curl http://localhost:8983/solr/update -H 'Content-type:application/json'
-d '[
 {"id"        : "book1",
  "copies_i"  : { "inc" : 1},
  "cat"       : { "add" : "fantasy"},
  "ISBN_s"    : { "set" : "0-380-97365-0"}
  "remove_s"  : { "set" : null } }
]'
```

A. Adding your own content to Solr

Dynamic fields

- Allow indexing of content with fields not mentioned in the schema
- Field names must match a pattern
- Pattern has a wildcard at the start or end:
  - `Store_*`
  - `*_s`
- Can be used to implement a schema-less index
  - Using the dynamic field definitions found in the default schema.xml

Segments

- Solr writes documents in groups, called segments
  - Segments do not change
  - Except a bit marking a document as deleted
- This can make updates expensive
- Segments can be merged
  - Manually (optimize), or automatically
  - Makes a new segment
  - Deleted documents removed

B. Deleting data from solr

How to delete all data

- remove the data directory
  - **$SOLR_HOME/server/solr/techproducts/data**
- execute a query
  - http://localhost:8983/solr/update?stream.body=<delete><query>*:*</query></delete>

Delete content by query

```
<delete>
  <id>ID01</id>
  <id>ID02</id>
</delete>

<delete>
  <query>category:electronics</query>
</delete>
```

C. Build a bookstore search

➢ Field attributes

- Type : int, flot, date, location, string, text
- Required
- Indexed
- Stored

➢ String versus text

- String is one term, even if multiple words
  - San Francisco
  - Not changed from what Solr received
- Text is usually split up into multiple terms
  - San
  - Francisco
  - May be considerably changed from what Solr received
  - The Running Shoes => run shoe

C. Build a bookstore search

➢ Fields in our data (material/bookData1.xml)

- author (type: text_general)
- category (type: descendent_path)
- description (type: text_general)
- isbn (type: string)
- numpages (type: int)
- price (type: float)
- publisher (type: text_general)
- pubdate (type: date)
- title (type: text_general)
- yearpub (type: string)
- store (type: location)
- id (type: string)

Lab 7. Build a bookstore search

1. Core 생성 : bin/solr create –c bookstore –d server/solr/configsets/basic_configs
2. $SOLR_HOME/server/solr/bookstore/conf/managed-schema를 텍스트 편집기로 열어서 이전 페이지에 정의된 필드 추가
3. 24페이지의 `descendent_path field type`을 managed-schema에 추가
4. name은 text 이고, type 은 text_general 인 필드를 추가한다.
5. 모든 필드를 text 필드에 복제하도록 아래와 같이 copyField를 추가한다.

```
<copyField source="author" dest="text"/>
<copyField source="category" dest="text"/>
<copyField source="description" dest="text"/>
……
```

6. Solr를 재시작한 후, material/bookData1.xml 파일을 bookstore core에 추가

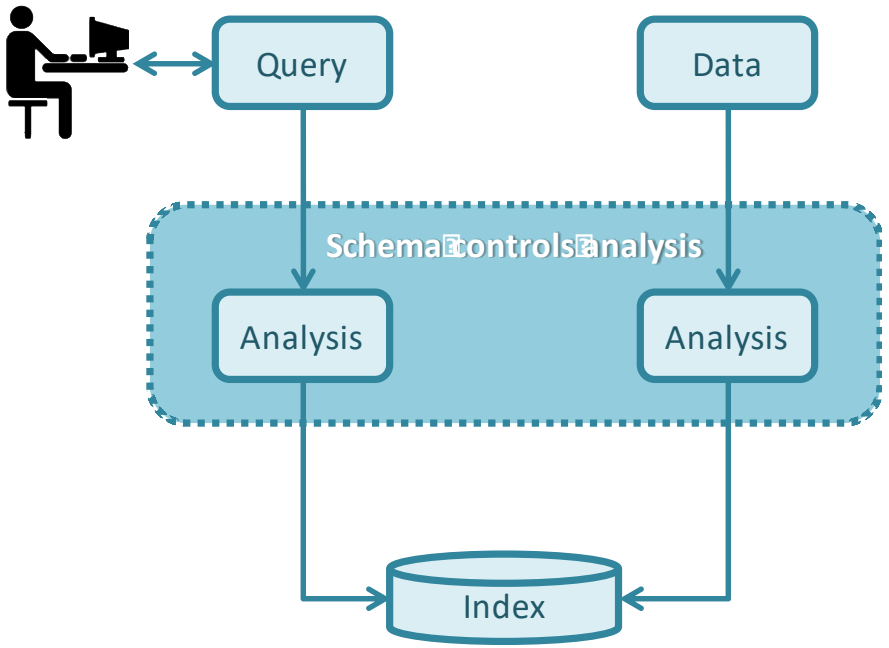
   bin/post –c bookstore ../material/bookData1.xml

7. http://localhost:8983/solr/#/bookstore 로 접속하여 색인 문서 건수 확인
8. Query 메뉴로 이동하여 category, yearpub으로 facet
9. Schema 메뉴로 이동하여 Fields 하위의 ISBN 선택
10. Click “Load Term Info” 하여 색인 데이터 확인

4. Updating your shcema

A. Adding fields to the schema

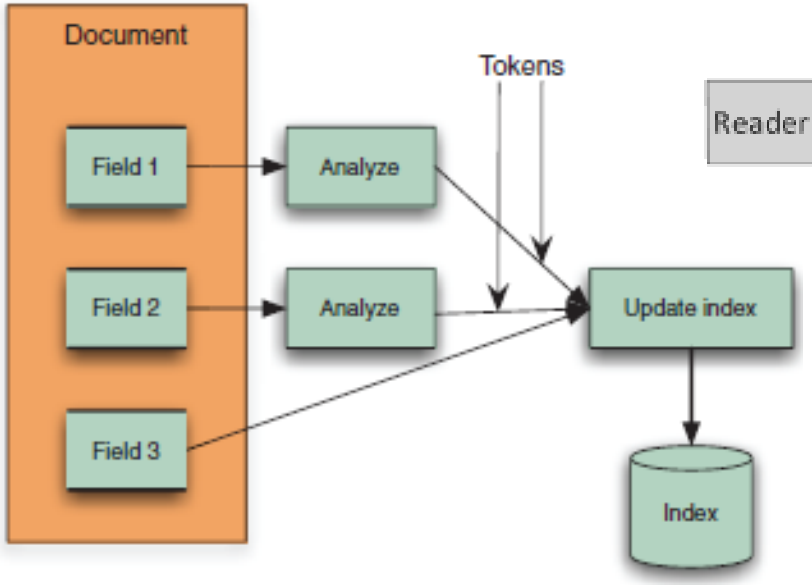
- Sections in managed-schema
  - Types
    - All types
    - Order doesn't matter
  - Fields
    - All fields (must have a type)
    - Order doesn't matter
  - Settings



B. Analyzing text

➢ Text Analysis 개요

- Text Analysis는 색인어를 추출하는 과정
- Text Analysis는 문장을 Term들로 변환하는 것
- Term = "필드명 + Token"
- Token은 색인어 추출, 소문자 치환, 불용어 제거 등의 과정을 거쳐 텍스트로부터 쪼개진 조각들



[Analysis Process]

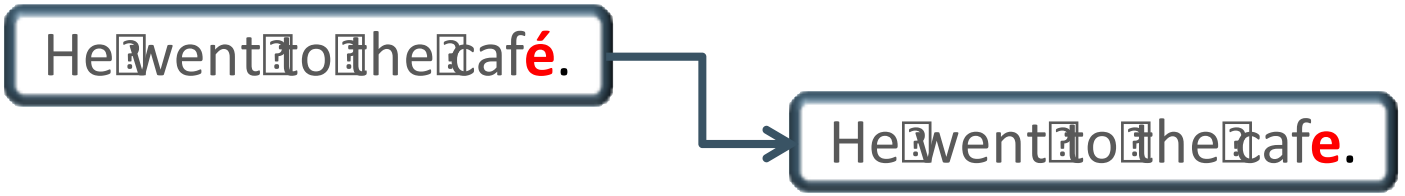
[Analyzer chain]

4. Updating your shcema

B. Analyzing text

➢ Character filters

```
<fieldType name="text_ws" class="solr.TextField">
  <analyzer>
    <charFilter
          class="solr.MappingCharFilterFactory"
        mapping="mapping-ISOLatin1Accent.txt"/>
    <tokenizer class="solr.WhitespaceTokenizerFactory"/>
    <filter class="solr.LowerCaseFilterFactory"/>
  </analyzer>
</fieldType>
```

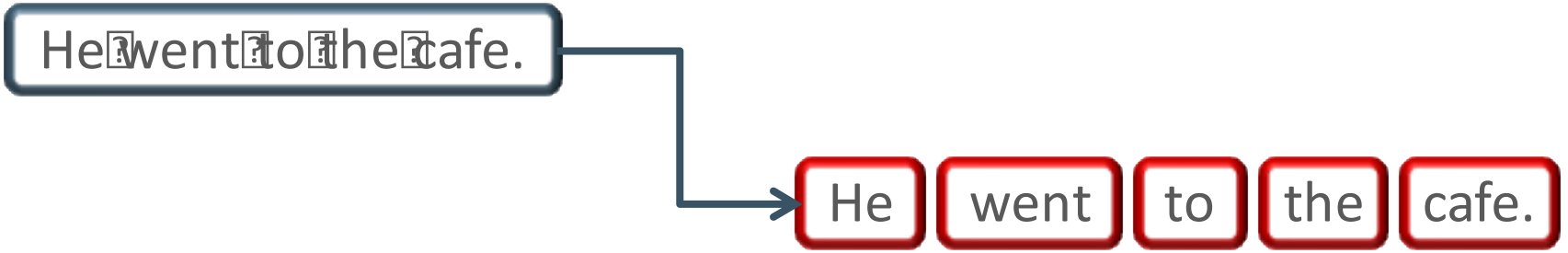


4. Updating your shcema

B. Analyzing text

➢ Tokenizers

```
<fieldType name="text_ws" class="solr.TextField">
  <analyzer>
    <charFilter
          class="solr.MappingCharFilterFactory"
        mapping="mapping-ISOLatin1Accent.txt"/>
    <tokenizer class="solr.WhitespaceTokenizerFactory"/>
    <filter class="solr.LowerCaseFilterFactory"/>
  </analyzer>
</fieldType>
```

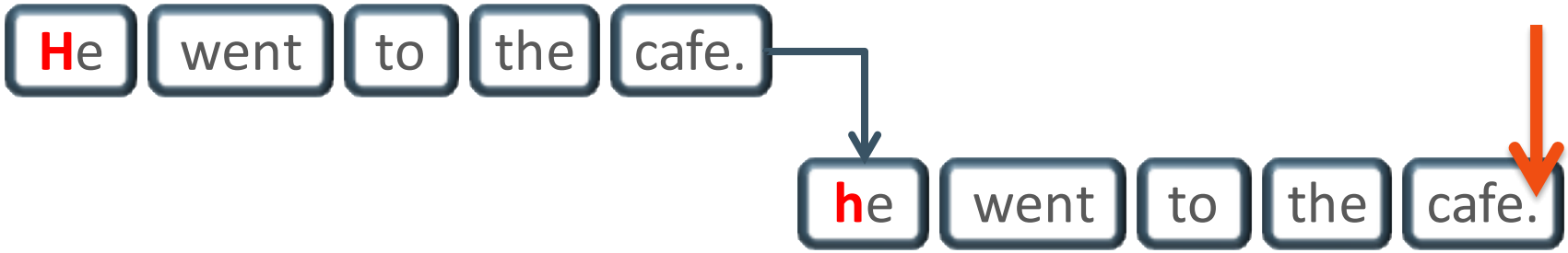


4. Updating your shcema

B. Analyzing text

➢ Token filters

```
<fieldType name="text_ws" class="solr.TextField">
  <analyzer>
    <charFilter
          class="solr.MappingCharFilterFactory"
        mapping="mapping-ISOLatin1Accent.txt"/>
    <tokenizer class="solr.WhitespaceTokenizerFactory"/>
    <filter class="solr.LowerCaseFilterFactory"/>
  </analyzer>
</fieldType>
```



4. Updating your shcema

C. Arirang Analyzer

➢ Download & 정보공유

http://café.naver.com/korlucene

➢ Source

https://github.com/soomyung

https://lucenekorean.svn.sourceforge.net/svnroot/lucenekorean

C. Arirnag Analyzer

➢ 형태소 분석이란?

- 형태소 분리
  - '의미가 있는 최소단위'인 형태소 분리

- 불규칙 원형 복원
  - 아름다운: 아름답+ㄴ, 걸어: 걷+어, 펐다: 푸+었+다

- 형태소의 품사 인식
  - 한국인명사+들접미사+은조사 전쟁명사 위험명사+을조사 느끼동사+지어미 않형용사+고어미 있형용사+다조사

출처 : 한국어 형태소 분석과 정보검색 (홍릉출판사, 강승식)

C. Arirnag Analyzer

➢ 형태소 분석 과정

- 입력 어절 사랑스러웠다
- 어미 분리 사랑스러웠 + 다
- 선어말 어미 사랑스러우 + 었 + 다
- 원형 복원 사랑스럽 + 었 + 다
- 접미사 분리 사랑 + 스럽 + 었 + 다

- 분리 순서: Left-to-right vs. Right-to-left

출처 : 한국어 형태소 분석과 정보검색 (홍릉출판사, 강승식)

C. Arirnag Analyzer

➢ 형태소 분석 과정 후보 : '가시는'

- 입력 어절 사랑스러웠다
- 어미 분리 사랑스러웠 + 다
- 선어말 어미 사랑스러우 + 었 + 다
- 원형 복원 사랑스럽 + 었 + 다
- 접미사 분리 사랑 + 스럽 + 었 + 다

- 분리 순서: Left-to-right vs. Right-to-left

출처 : 한국어 형태소 분석과 정보검색 (홍릉출판사, 강승식)

C. Arirnag Analyzer

➢ 형태소 분석 과정 후보 : '가시는'

1. (V "가")<lgV:18> + (f "시") + (e "는")
2. (V "갈")<T:18> + (f "시") + (e "는")
3. (V "가시")<lT:20> + (e "는")
4. (N "가시")<N:20> + (j "는")<1>

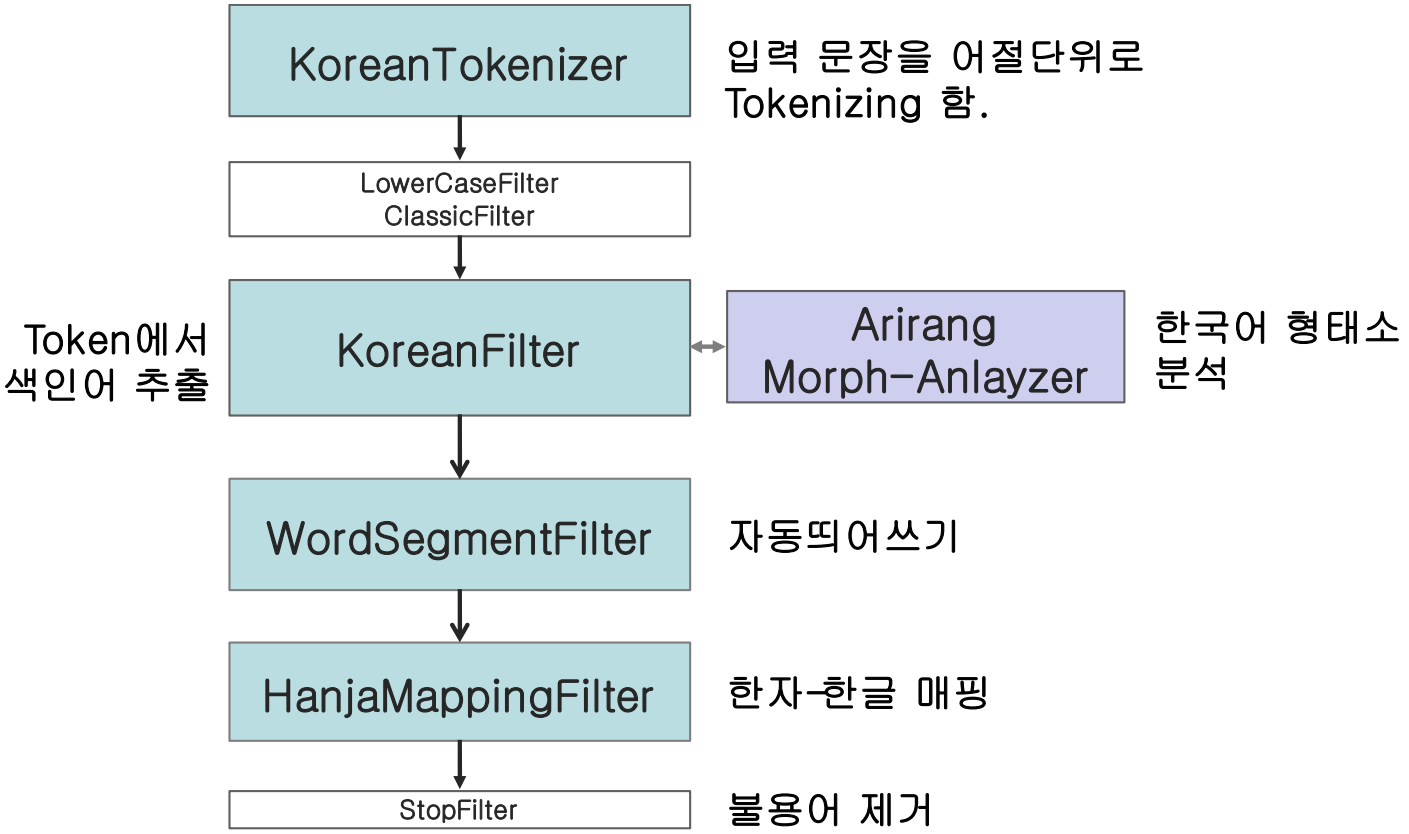
5. (V "가실")< :100> + (e "는")
6. (N "가시느")< :100> + (j "ㄴ")<1>
7. (N "가시늘")< :100> + (j "ㄴ")<1>
8. (N "가시늏")< :100> + (j "ㄴ")<1>

출처 : 한국어 형태소 분석과 정보검색 (홍릉출판사, 강승식)

4. Updating your shcema

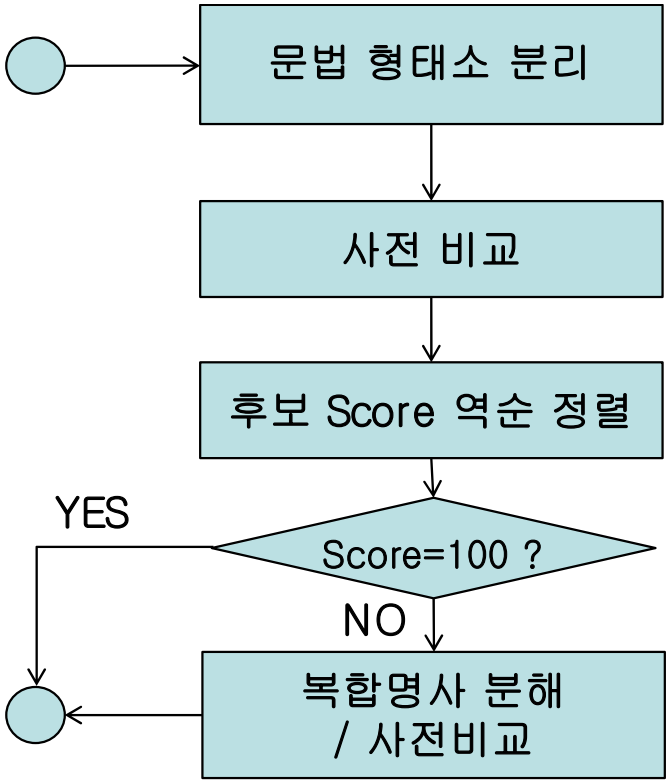
C. Arirang Analyzer

➢ Arirang Analyzer 구조



C. Arirang Analyzer

➢ 형태소 분석 흐름



사전과 비교하여 Scoring
(사전 존재 : Score = 100)

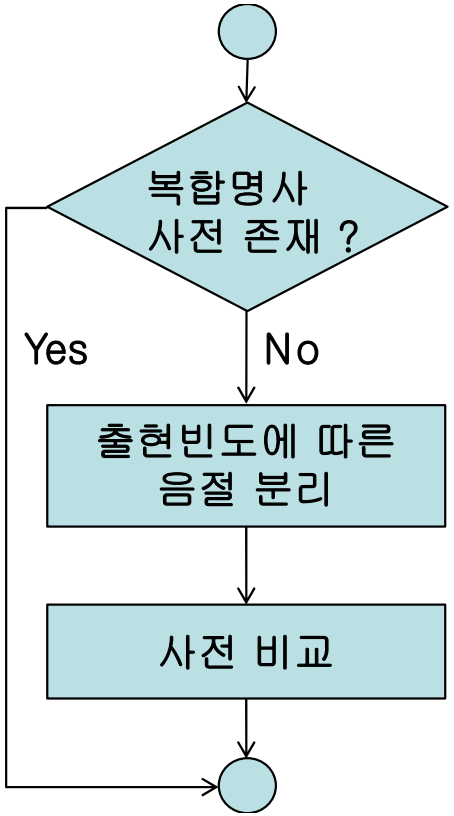
분석 성공 여부 점검

사전에 부분 존재 (Score = 70)
사전에 전체 존재 (Score = 100)

4. Updating your shcema

C. Arirang Analyzer

➢ 복합명사 분해



복합명사 음절분리 규칙

1. 3음절 복합명사
   2음절 + 1음절 / 1음절 + 2음절
2. 4음절 복합명사
   1음절 + 3음절 / 1음절 + 2음절 + 1음절 / 2음절 + 2음절
3. 5음절 복합명사
   2음절 + 3음절 / 3음절 + 2음절 / 4음절 + 1음절 / 2음절 + 2음절 + 1음절 / 2음절 + 1음절 + 2음절
4. 6음절 이상 복합명사
   최장 명사를 기준으로 좌.우를 분리하여 1~4번 적용

C. Arirang Analyzer

➢ 사전구성

1. total.dic : 기본사전
2. extension.dic : 확장사전
3. josa.dic : 조사사전
4. eomi.dic : 어미사전
5. prefix.dic : 접두어 사전
6. suffix.dic : 접미어 사전
7. compounds.dic : 기분석 복합명사 사전
8. syllable.dic : 음절정보사전

C. Arirang Analyzer

➢ 사전 Customizing

- 사전위치 : org.apache.lucene.analysis.kr.dic
- 사전 찾는 순서
  1. Application Classpath 의 패키지 경로에 있는 사전
  2. KoreanAnalyzer 의 Jar 파일에 내장된 사전
- 사전 Customizing 방법
  - extension.dic 를 Application classpath 에 위치 (추천)
  - total.dic 를 Application classpath 에 위치 (추천하지 않음)

C. Arirang Analyzer

➢ 사전 구성

○ 기본 및 확장사전 구성

```
사랑,100100000X
```

콤마(,)를 중심으로 좌측은 단어, 우측은 단어정보

○ 단어정보 의미

```
 1   2      3        4          5        6 7 8 9      10
체언 용언  기타  하여동사  되어동사   reserved  불규칙변형
```

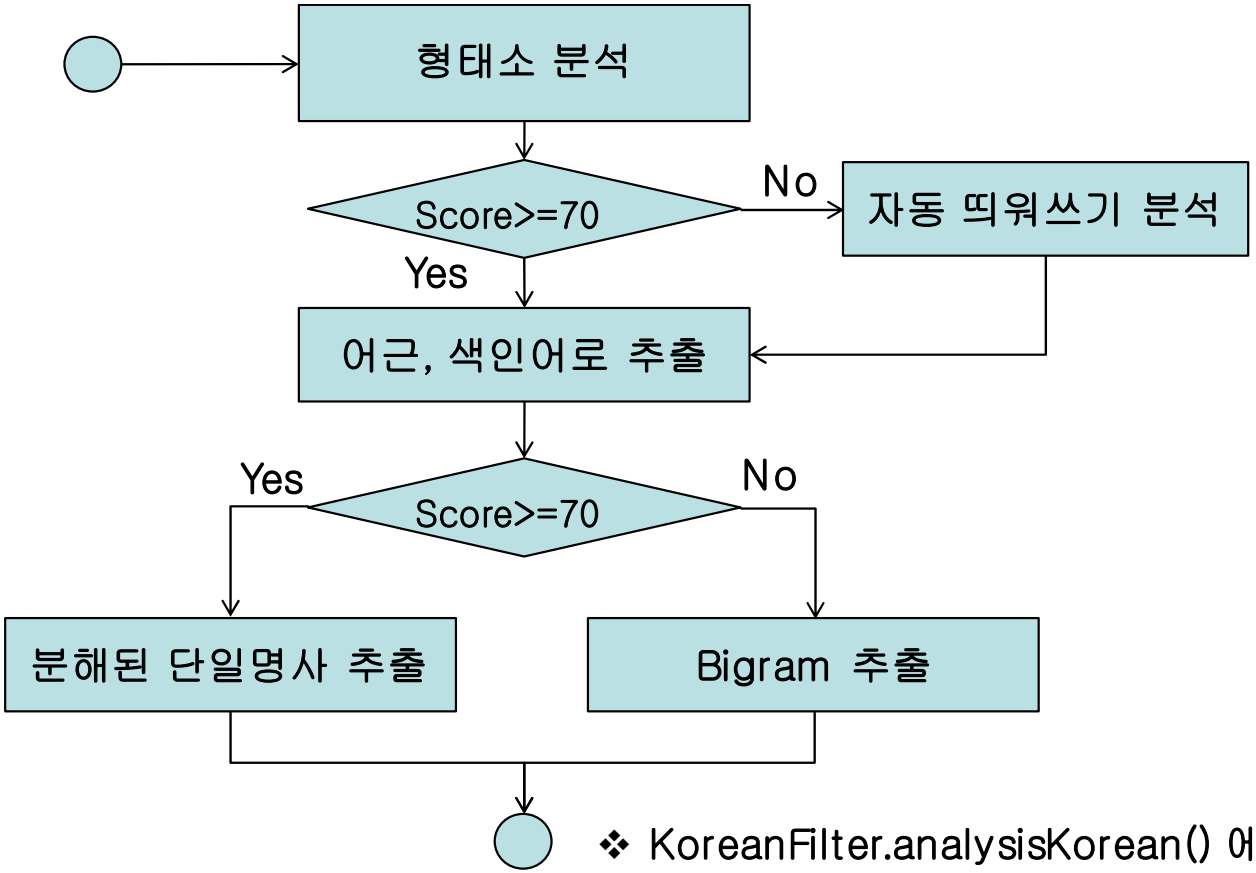
- 불규칙변형의 종류

  B:ㅂ 불규칙, H:ㅎ 불규칙, L:르 불규칙, U:ㄹ 불규칙, S:ㅅ 불규칙, D:ㄷ 불규칙,

  R:러 불규칙, X:규칙

C. Arirang Analyzer

➢ 한글색인어 추출

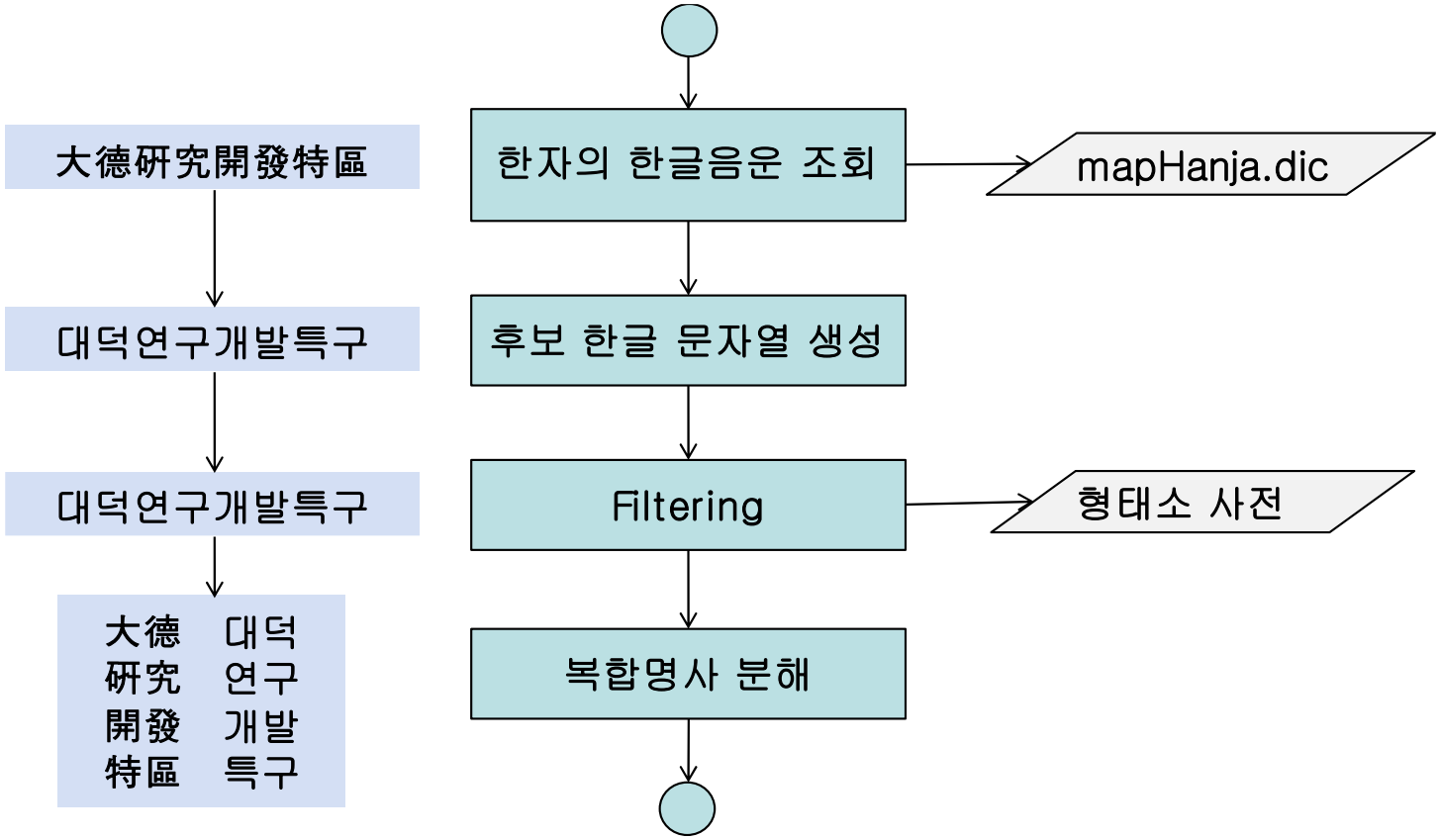


❖ KoreanFilter.analysisKorean() 에서 처리

4. Updating your shcema

C. Arirang Analyzer

➢ 한자 색인어 추출



4. Updating your shcema

C. Arirang Analyzer

➢ Solr schema 설정

```
<fieldType name="ko" class="solr.TextField">
  <analyzer type="index">
     <tokenizer class="org.apache.lucene.analysis.ko.KoreanTokenizerFactory"/>
        <filter class="solr.LowerCaseFilterFactory"/>
        <filter class="solr.ClassicFilterFactory"/>
        <filter class="org.apache.lucene.analysis.ko.KoreanFilterFactory"
                  queryMode="false" hasOrigin="true" hasCNoun="true" bigrammable="false"/>
        <filter class="org.apache.lucene.analysis.ko.HanjaMappingFilterFactory"/>
        <filter class="org.apache.lucene.analysis.ko.PunctuationDelimitFilterFactory"
                 hasConcatedTerm="false"/>
        <filter class="solr.StopFilterFactory" words="stopwords.txt" ignoreCase="true"/>
   </analyzer>
   <analyzer type="query">
     <tokenizer class="org.apache.lucene.analysis.ko.KoreanTokenizerFactory"/>
     <filter class="solr.LowerCaseFilterFactory"/>
     <filter class="solr.ClassicFilterFactory"/>
     <filter class="org.apache.lucene.analysis.ko.KoreanFilterFactory" queryMode="true"
                hasOrigin="false" hasCNoun="true" bigrammable="false"/>
     <filter class="org.apache.lucene.analysis.ko.HanjaMappingFilterFactory"/>
     <filter class="org.apache.lucene.analysis.ko.PunctuationDelimitFilterFactory"
            hasConcatedTerm="false"/>
     <filter class="solr.StopFilterFactory" words="stopwords.txt" ignoreCase="true"/>
    </analyzer>
</fieldType>
```

Lab 8. Analyzing Text

1. $SOLR_HOME/server/solr/bookstore/conf/managed-schema를 텍스트 편집기로 연다
2. 이전 페이지의 ko fieldType을 추가한다.
3. material/ arirang.lucene-analyzer-6.1-1.0.0.jar, arirang-morph-1.1.0.jar을 $SOLR_HOME/server/solr/lib에 복사한다.
4. type이 text_general 인 모든 필드의 type을 ko 로 교체한다.
5. Solr를 재시작 후, bookData를 재색인한다.

   bin/post -c bookstore../material/bookData*
6. “한강”으로 검색하여 몇건이 검색되었는지 확인한다.
7. Analysis 메뉴로 이동한다.
8. Field Value (Index)에 “여자의 열매”를 입력하고, Analyse Fieldname / FieldType에서 ko를 선택한 후 “Analyse Values”를 클릭하여 추출된 색인어를 확인한다.
9. Field Value (query)에 “여자의 열매”를 입력하고, Analyse Fieldname / FieldType에서 ko를 선택한 후 “Analyse Values”를 클릭하여 추출된 색인어를 확인한다.

5. Relevance

A. Field weighting

- What's most important?
  - Title:potter
  - Author:potter
  - Description:potter
  - Reviews:potter

- Syntax
  - standard: title:potter^10 author:potter^5 description:potter
  - dismax/edismax: title^10 author^5 description

5. Relevance

B. Phrase queries

- Which is a better match?
  - Query: "harry potter"
  - Text:
    - Harry was a nice man. He lived on main street, next door to a potter.
    - Harry Potter was a wizard
- Why?
- Phrase boosting parameter
  - pf, pf2, pf3
  - ps, ps2, ps3

5. Relevance

C. Function queries

- So far: text matching
- What about non-text factors?
  - price, distance, date
- Reference
  - https://wiki.apache.org/solr/FunctionQuery

5. Relevance

D. Fuzzy and wildcard search

- Sometimes you don't know exactly what you are looking for
  - It starts with pro: pro*
  - It ends with tion: *tion
  - I'm not sure of the second letter: c?t
  - It's something like steve:
    - steve~
    - steve~0.9
  - It matches a regular expression:
    - /Ap.*e/ matches Apache

6. Extended features

A. More-like-this

- Finds similar documents, based on
  - The contents of document(s) in the index
  - Data provided as a parameter
- Builds & runs a query
- Try blow query
  - http://localhost:8983/solr/techproducts/browse?&q=name:ipod&mlt=true&mlt.fl=name&mlt.qf=name&mlt.boost=true&mlt.mintf=1&mlt.mindf=1

6. Extended features

B. Geospatial

- Multiple values per field
- Index shapes other than points
  - circles, polygons, etc.
- Well Known Text (WKT) support via JTS
- Indexing:

```
"geo":"43.17614,-90.57341"
"geo":"Circle(4.56,1.23 d=0.0710)"
"geo":"POLYGON((-10 30, -40 40, -10 -20, 40 20, 0 0, -10 30
))"
```

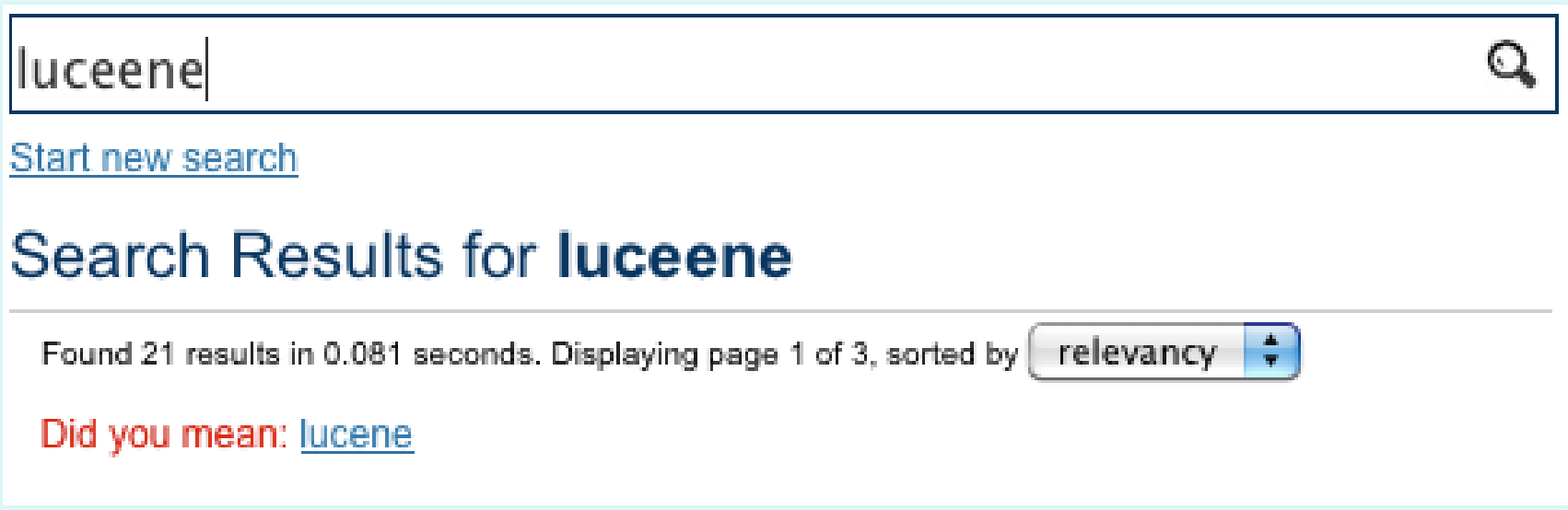
- Searching:

```
fq=geo:"Intersects(-74.9 41.4 -69.4 44.5)"
fq=geo:"Intersects(POLYGON((-10 30, -40 40, -10 -20, 40 20,
0 0, -10 30)))"
```

6. Extended features

C. Spell checking

- Improves findability
- Can build from index or file



6. Extended features

D. Suggestion

- Improves findability
- Can build from index or file
  - Shares code with spellcheck

6. Extended features

E. Highlighting

```
  <int name="f.price.facet.range.end">600</int>
  <int name="f.price.facet.range.gap">50</int>
  <str name="facet.range">popularity</str>
  <int name="f.popularity.facet.range.start">0</int>
  <int name="f.popularity.facet.range.end">10</int>
  <int name="f.popularity.facet.range.gap">3</int>
  <str name="facet.range">manufacturedate_dt</str>
  <str name="f.manufacturedate_dt.facet.range.start">NOW/YEAR-10YEARS</str>
  <str name="f.manufacturedate_dt.facet.range.end">NOW</str>
  <str name="f.manufacturedate_dt.facet.range.gap">+1YEAR</str>
  <str name="f.manufacturedate_dt.facet.range.other">before</str>
  <str name="f.manufacturedate_dt.facet.range.other">after</str>


  <!-- Highlighting defaults -->
  <str name="hl">on</str>
  <str name="hl.fl">title author description publisher</str>
  <str name="hl.useFastVectorHighlighter">true</str>
  <str name="hl.simple.pre">&lt;font color="red"&gt;</str>
  <str name="hl.simple.post">&lt;/font&gt;</str>
  <str name="f.name.hl.fragsize">0</str>
  <str name="f.name.hl.alternateField">name</str>
</lst>
<arr name="last-components">
 <str>elevator</str>
  <str>spellcheck</str>
</arr>
<!--
<str name="url-scheme">httpx</str>
-->
```

6. Extended features

F. Pseudo-fields

- Returns other info along with document stored fields
  - Function queries

    ```
    fl=name,location,geodist(),add(myfield,10)
    ```
  - Fieldname globs

    ```
    fl=id,attr_*
    ```
  - Multiple "fl" (field list) values

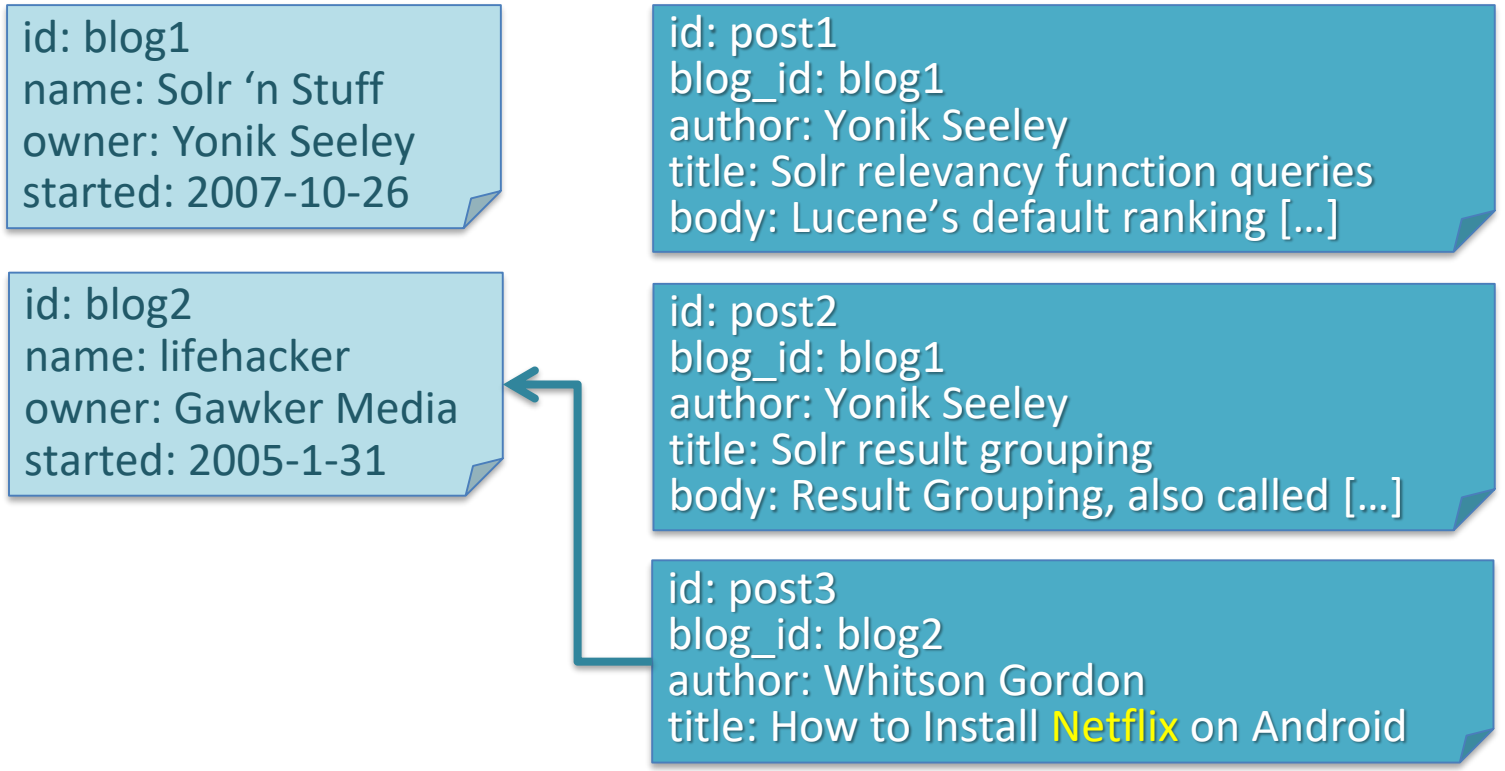
    ```
    &fl=id,attr_*
    &fl=geodist()
    &fl=termfreq(text,'solr')
    ```
  - Aliasing

    ```
    fl=id,location:loc,_dist_:geodist()
    fl=id,[explain],[shard]
    ```

G. Pseudo-joins (1/4)

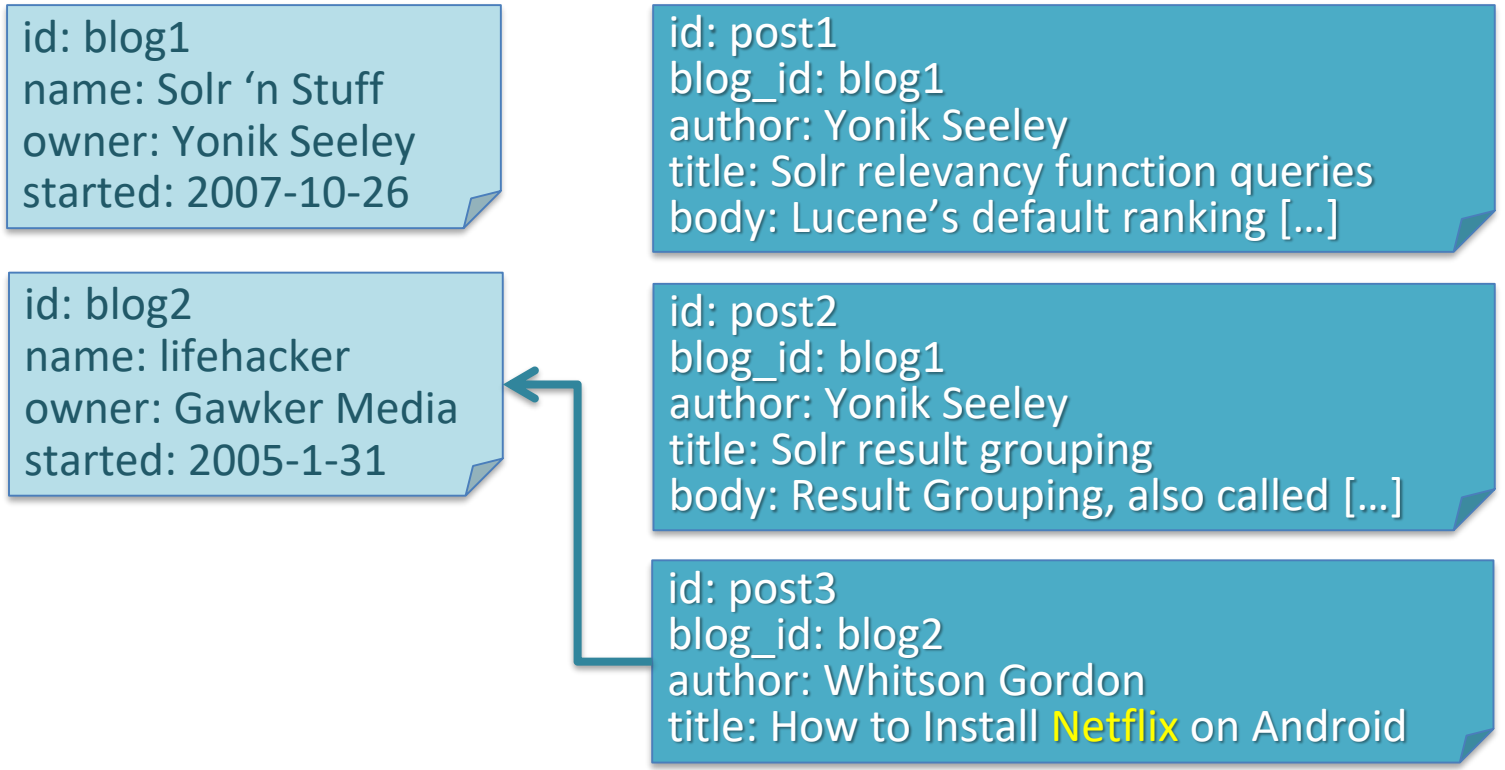
- Restrict to blogs mentioning netflix:

```
fq={!join from=blog_id to=id} body:netflix
```



G. Pseudo-joins (2/4)

- How it works:
  - Finds all documents matching `netflix`
  - Maps to different docs by following `blog_id` to `id`



6. Extended features

G. Pseudo-joins (3/4)

- Only show posts from blogs started in 2010 or after

```
&fq={!join from=id to=blog_id}started:[2010 TO *]
```

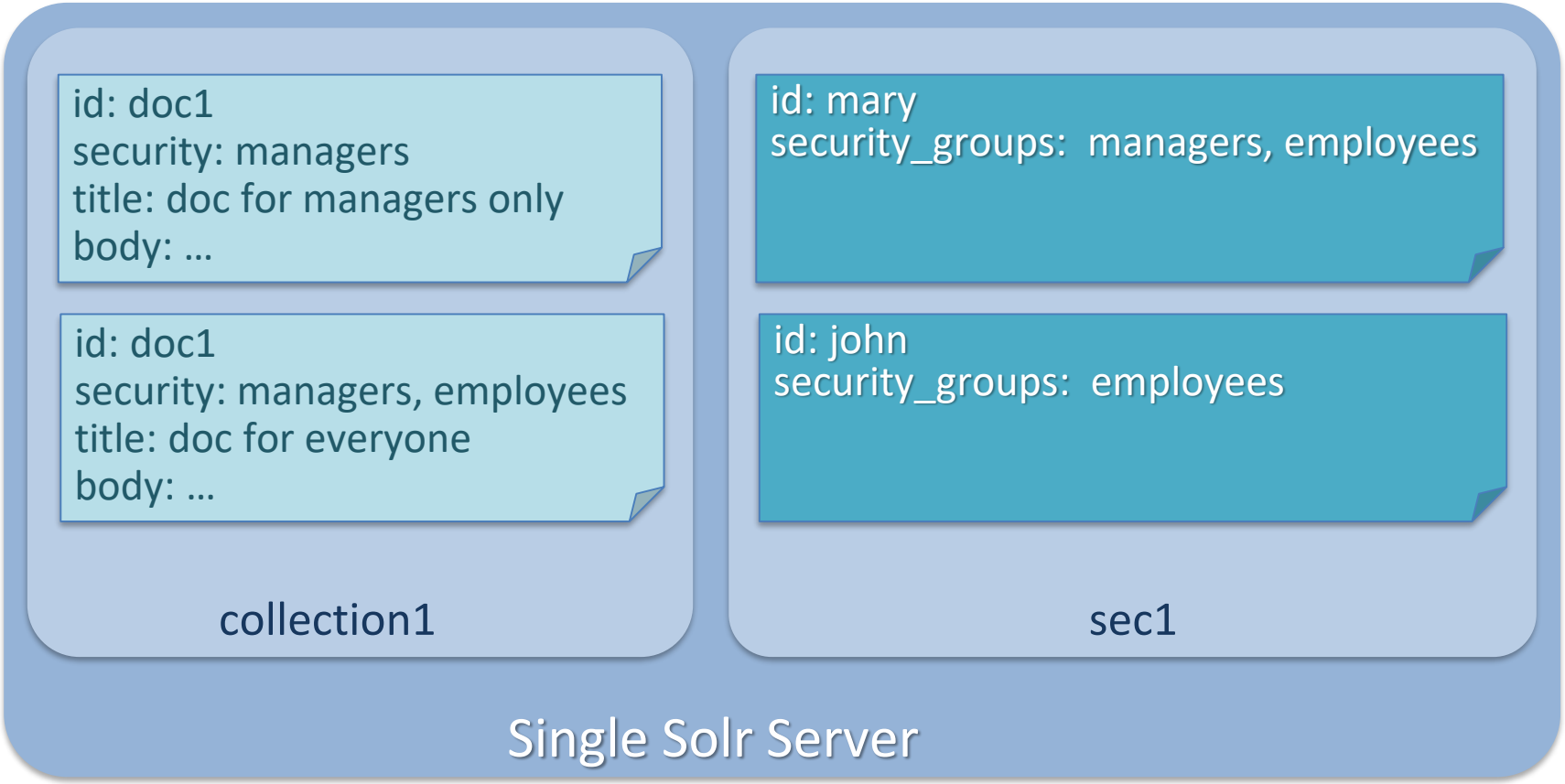
- If a post in a blog mentions "embassy", search all posts in that blog f or "bomb" (self-join)

```
q=bomb
&fq={!join from=blog_id to=blog_id}embassy
```

- If a blog post mentions "embassy", search all emails with the same bl og owner for "bomb"

```
q=email_body:bomb &fq=
{!join from=owner_email_user to=email_user} {!join from=blo
g_id to=id}embassy
```

G. Pseudo-joins (4/4)




```
http://localhost:8983/solr/collection1/select?q=foo
&fq={!join fromIndex=sec1 from=security_groups
to=security}user:john
```

7. Data Import Handler

A. Introduction

- Imports data from RDBMS/XML into Solr using configuration
- Import works across multiple tables
- Data is denormalized
- Supports full and incremental update
- Allows to plugin components
- Is a contrib module

B. Import

Full Import

- Indexes complete data to Solr
- command=full-import
- Updates the dataimport.properties

Delta Import

- Incremental Update
- command=delta-import
- Tables require additional column last_modified timestamp
- Relies on dataimport.properties file, which keeps the last indexed time

C. Configuration Steps

- Configure the DIH in SolrConfig.xml

```
<requestHandler name="/dataimport"
      class="org.apache.solr.handler.dataimport.DataImportHandler">
   <lst name="defaults">
    <str name="config">/path/to/data-config.xml</str>
   </lst>
</requestHandler>
```

- Configure the datasource and data-config.xml
- Add the dependencies to the lib directory

E. Extending the API

- Transformers
- EntityProcessors
- DataSource
- EventListeners

F. Transformers

- Modifies the value of a field or creates a new field altogether
- Transformers can be chained
- Some built-in transformer:
  - RegexTransformer
  - DateFormatTransformer
  - TemplateTransformer
  - HTMLStripTransformer

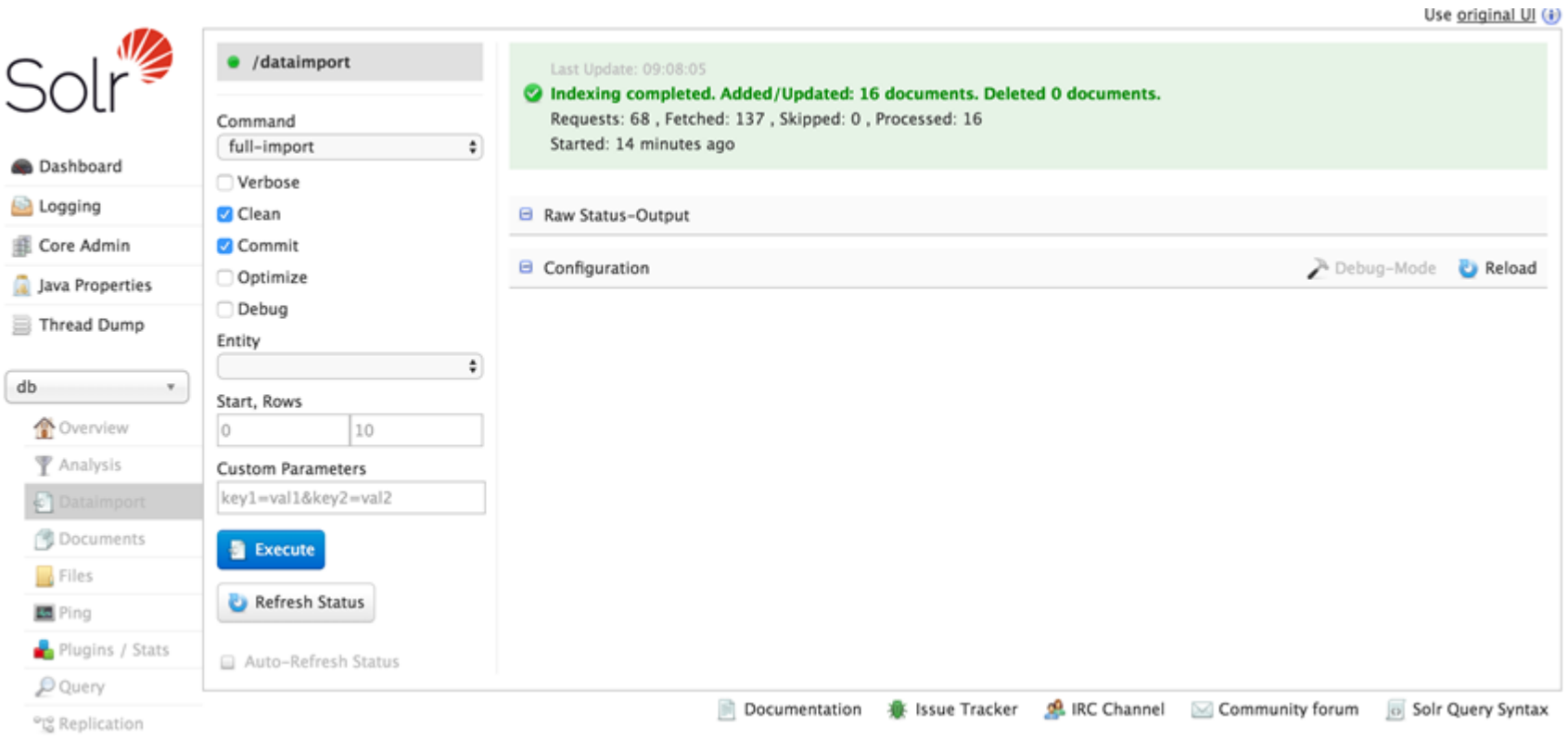
F. Transformers

- Modifies the value of a field or creates a new field altogether
- Transformers can be chained
- Some built-in transformer:
  - RegexTransformer
  - DateFormatTransformer
  - TemplateTransformer
  - HTMLStripTransformer

7. Data Import Handler

G. Demo

- bin/solr start –s example/example-DIH/solr



8. SolrCloud

A. Introduction

- SolrCloud combines fault tolerance and high availability
- Central configuration for the entire cluster
- Automatic load balancing and fail-over for queries
- ZooKeeper integration for cluster coordination and configuration.

B. How SolrCloud works

- Indexing
  - There really are no masters/slaves in SolrCloud
  - Instead, there are leaders and replicas
    - Leaders are automatically elected
    - If a leader goes down, one of its replicas is automatically elected as the new leader

- Searching
  - Searching just happens
    - No distinction between masters and slaves
    - A request can be sent to any machine in the cluster
  - Searching is NRT
    - Replication deprecated; distributed indexing instead
    - Small delay while docs are forwarded to replicas
  - No need to specify a shards parameter in solrconfig.xml

C. SolrCloud Example

```
$ bin/solr -e cloud

Welcome to the SolrCloud example! This interactive session
will help you launch a SolrCloud cluster on your local
workstation. To begin, how many Solr nodes would you like to
run in your local cluster? (specify 1-4 nodes) [2]

Please enter the port for node1 [8983]

Please provide a name for your new collection:
[gettingstarted]
```

D. Start & Stop

- Stop

  bin/solr stop -all

- Start

  - Start the first node

    bin/solr start –cloud –s example/cloud/node1/solr

  - Start the rest of nodes

    bin/solr start –cloud –s example/cloud/node2/solr –p 7574 –z localhost:9983

E. Manage Collections

- Delete collection

  bin/solr delete –c $collectionName –deleteConfig true

- Create collection

  bin/solr –c $collectionName –d $config-dir –s $numOfShards