



Solr – Soup to Nuts*

October 7, 2009

Clay Webster

Scope

- » Solr the Product
- » Why Make Solr?
- » Building Open Source within a Business
- » How to Use Solr





Solr Basics

The Obligatory “What is Solr?”

- » Solr is a scalable enterprise software platform that provides highly relevant full text search with “faceting”.
- » Solr is the defacto implementation of the Lucene Java search library.
- » Solr is easy to set up. To get powerful features you do not need to write code. Just configure the schema and word analyzers and craft powerful queries.
- » Solr is built to be blindingly fast, reliable, and capable of intense operational demands – many caching, replication, and administration controls.
- » Solr is internally programmable and has restful APIs.
- » Solr is open source, actively supported and 100% free.



Solr Features



» High Relevancy – the right documents with the right search scores. Index schema, word analyzers, search queries, are all easy ways to enhance relevancy.



» Search with Faceted Drill-downs – users narrow their search results to a guaranteed number of results in.



» Open Source – Solr is free, open source software. A thriving development community at the Apache Software Foundation is constantly adding new features to Solr, ensuring it will be around for a long time.



» Fast – Solr is blindingly fast and will not be a bottleneck in your content delivery.





Why Make Solr?

Sticker-Shock

- » All Initial Vendor Pricing Far Higher Than Budgeted
- » Not Site License Pricing
- » Vendors Use Metrics Like #CPUs That Would Result in >\$1M.
- » Some Prices by Queries/Sec ...
- » Vendors' Prices aren't Apples to Apples

2005 Vendor Pricing

Vendor	Model	Interested Metrics	Initial Proposed Price/Formula	Rough Initial Cost
F*****	Queries/sec, Doc Space, Tools, Maint	Queries/sec, Num Docs, Size Docs, Doc Space	\$627,000	\$627K
C*****	CPUs (a little complex) and Tools	CPUs	\$287,000 + (\$75,000 * CPU) + Build (Millions)	\$2,870K
E*****	not yet received	Licensed Term and based on Query-side CPUs	Siloization puts us off their chart. > \$750,000	\$900K
A*****	Interfaces, distrib requirements, perf requirements, failover requirements, query demand	Queries/sec, redundancy reqs, connectors, functionality, num documents, ballpark anonymous users, world-wide-web user license	\$500,000 (but I'm guessing larger)	\$500K
R*****	CPUs	CPUs	\$75,000 * CPU (Millions)	\$2,000K
E*****	Enterprise Models (plural), with some PR	Base it on the business	\$150,000 => \$400,000	\$400K
I*****	CPUs (a little complex) and Internal User License	CPUs, Internal Users	\$340,000 + (\$75,000 * CPU) + Build (Millions)	\$2,000K
V*****	Pretty much, "no".	CPUs, intranet users, content size, sources, response time, training, consulting, etc.	~\$200,000, with larger ones in high six figures.	\$800K

What To Do?

- » Search for a Replacement Search Platform
 - commercial: high license fees
 - open-source: no good solutions
- » **FAIL**



Early (aka 2006) Solr Credits

- » Ted Cahall
 - » Mark Castrovinci
 - » Clay Webster
 - » Yonik Seeley
 - » Bill Au
 - » Chris Hostetter
- (Not necessarily in that order.)



Lucene “Refresher”

- » Lucene is a search library
- » Add documents to an index via `IndexWriter`
 - A document is a collection of fields
 - No config files, dynamic field typing
 - Flexible text analysis – tokenizers, filters
- » Search for documents via `IndexSearcher`
 - Hits = `search(Query, Filter, Sort, topN)`
- » Scoring is modified `tf*idf`
 - term frequency–inverse document frequency



Solr Architecture

- » Basics
 - Java, Lucene, Appserver
- » Queries
 - Query Language
 - HTTP GET requests with XML, Ruby, PHP, JSON responses
 - HTTP POSTs of XML documents
 - Custom Query Handlers
- » Data Architecture and Schema
 - Types, Analysis, Tokenizers, Dynamic Fields
- » Service Tier(s)
 - Master & Query Server
- » Data Loading
 - Request/Update Handlers and Response Writers, Analyzers, Plugin Architecture, CSV Loader, Java, DB sync
- » Distribution
- » Administration





Building Open Source
within a Business

Why Open Source Solr? (Development)

- » New concepts not of our origin
 - Useful features
 - Not currently useful features
 - Questionable features
- » Extra developer power
 - Optimizations
 - Features we didn't have time to do
 - Expertise
- » Bonded to Lucene



Why Open Source Solr? (Software Quality)

- » Quality contributions/commits
 - Embarrassment factor
- » Code and functionality reviews
 - Normal course of business
- » Productization
 - More than normal
- » Collaborative Design – checks and balances



Why Open Source Solr? (Bugs!)

- » Bugs Found
 - More than CNET would find
 - Bugs before CNET hits
 - Bugs CNET would never hit
- » Bugs Fixed
 - Fixing bugs we care about
 - Fixing bugs we don't care about
 - Receiving contributed fixes
 - Reviewing committed fixes
- » Far less bugs



Why Not Open Source Solr?

- » Engineer Time
 - Upfront packaging
 - Process
 - Leadership
- » Dumb user questions
- » Split infrastructure
 - Website, wiki, mailing lists, build systems, repository, bug systems
- » Red tape
 - Procedural stuff aimed at making Apache projects successful
- » Far less control than internal products
 - Voting/Consensus



Solar vs. Solr

- » Solar?
 - History of architecturally descriptive acronyms
 - Themed from ATOMICS (Apache TO Mysql In Cnet Search)
 - Solar == Search on Lucene and Resin
 - Solar is also in a Lucene light theme
- » Couldn't Use "Solar"
- » Rename to a non-acronym?



Solar vs. Solr

» Related “Light” Names

Name	Meaning	Origin	Gender	Visual Effect
Alena	Light	Slavic	Female	alena-dev@lucene.apache.org
Brigid	Bringer/Bearer of Light	Celtic/Gaelic	Female	brigid-dev@lucene.apache.org
Huda	Enlightenment, Guidance	Arabic	Female	huda-dev@lucene.apache.org
Ilona	Beautiful Sunshine	Hungarian	Female	ilona-dev@lucene.apache.org
Inara	Ray of Light - Heaven-Sent	Arabic	Female	inara-dev@lucene.apache.org
Leyna	Bright and Shining Light	Russian	Female	leyna-dev@lucene.apache.org
Luca	Bringer of Light	Italian	Either	luca-dev@lucene.apache.org
Luce	Light	Latin	Male	luce-dev@lucene.apache.org
Lucine	Bright, Light	French	Male	lucline-dev@lucene.apache.org
Lucius	Bringer of Light	Latin	Male	lucius-dev@lucene.apache.org
Misae	White Hot Sun	Native-American	Either	misae-dev@lucene.apache.org
Morag	Embracing The Sun	Celtic/Gaelic	Female	morag-dev@lucene.apache.org
Sulwyn	Bright As The Sun	Welsh	Female	sulwyn-dev@lucene.apache.org
Synnove	Sun Gift	Scandinavian	Female	synnove-dev@lucene.apache.org
Lyuaes	An epithet of Dionysys, as the god who releases people from worries	Greek	?	lyuaes-dev@lucene.apache.org

<http://www.pantheon.org/>
<http://www.babynames.com/>



History

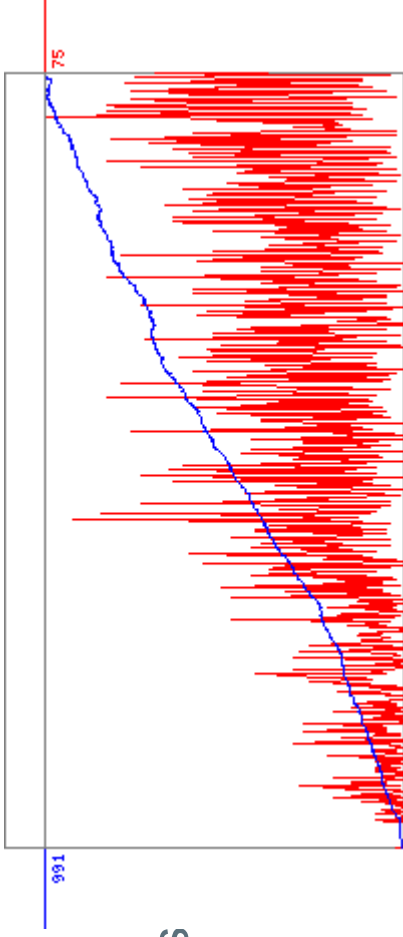
- » CNET grants code to Apache
- » Solr enters Incubator 17 Jan 2006
- » Solr is a Lucene sub-project
- » Production: CNET Reviews, CNET Channel, Shopper.com, News.com, Download.com, ChowHound



Community Activity (last 975 days)

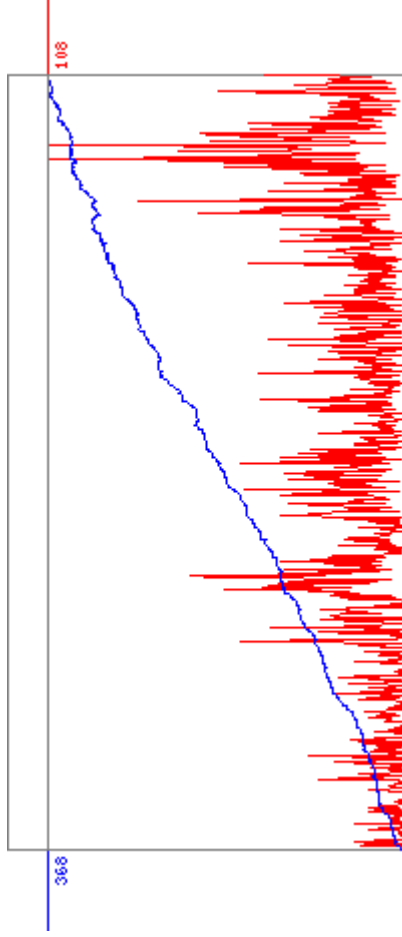
» “User” List:

- 1113 Current Subscribers
- Posts: 15343
- Avg posts per day: 15.74



» Developer List:

- 375 Current Subscribers
- Posts: 12543
- Avg posts per day: 12.86



What Did We Get From Open Sourcing?

- » Longevity
 - Well-received, growing community
 - Solr is the open source search software that people use and add to
 - Open source software is often preferred
- » Collaboration
 - Voting/Consensus
 - Opposite side of losing control
- » Leadership
 - Driving position vs. passenger user/customer position
 - Solr keeps in sync with CNET needs
 - CNET gets the best support of all Solr users
- » Positive karma
 - Recruiting
 - Industry respect
- » OOTB goodness





Using Solr — content
scraped from Yonik and Chris

Adding Documents

HTTP POST to /update

```
<add><doc boost="2">  
  <field name="article">05991</field>  
  <field name="title">Apache Solr</field>  
  <field name="subject">An intro...</field>  
  <field name="category">search</field>  
  <field name="category">lucene</field>  
  <field name="body">Solr is a full...</field>  
</doc></add>
```



Deleting Documents

» Delete by Id

```
<delete><id>05591</id></delete>
```

» Delete by Query (multiple documents)

```
<delete>
```

```
<query>category:lucene</query>
```

```
</delete>
```



Commit

- » `<commit/>` makes changes visible
 - closes `IndexWriter`
 - removes duplicates
 - opens new `IndexSearcher`



Searching: Query Syntax

Lucene Query Syntax

- » mission impossible
- » +mission +impossible –actor:cruise
- » “mission impossible” –actor:cruise
- » title:spiderman^10 description:spiderman
- » description:“spiderman movie”~10
- » +HDTV +weight:[0 TO 100]
- » Wildcard queries: te?t, te*t, test*



Querying Data

» HTTP GET or POST, params specifying query options...

`http://solr/select?q=electronics`

`http://solr/select?q=electronics&sort=price+desc`

`http://solr/select?q=electronics&rows=50&start=50`

`http://solr/select?q=electronics&fl=name+price`

`http://solr/select?q=electronics&fq=inStock:true`



Searching: Parameters

Query Arguments for HTTP GET/POST to /select

param	default	description
q		The query
start	0	Offset into the list of matches
rows	10	Number of documents to return
fl	*	Stored fields to return
qt	standard	Query type; maps to query handler
df	(schema)	Default field to search

Querying Data: Results

» Canonical response format is XML...

```
<response>  
  <lst name="responseHeader">  
    <int name="status">0</int>  
    <int name="QTime">1</int>  
  </lst>  
  <result name="response" numFound="14" start="0">  
    <doc>  
      <arr name="cat">  
        <str>electronics</str>  
        <str>connector</str>  
      </arr>  
      <arr name="features">  
        <str>car power adapter, white</str>  
      </arr>  
      <str name="id">F8V7067-APL-KIT</str>  
    </doc>  
  </result>  
</response>
```



Caching

IndexSearcher's view of an index is fixed

- Aggressive caching possible
- Consistency for multi-query requests

filterCache – unordered set of documents matching a query

resultCache – ordered subset of documents matching a query

documentCache – the stored fields of documents

userCaches – application specific

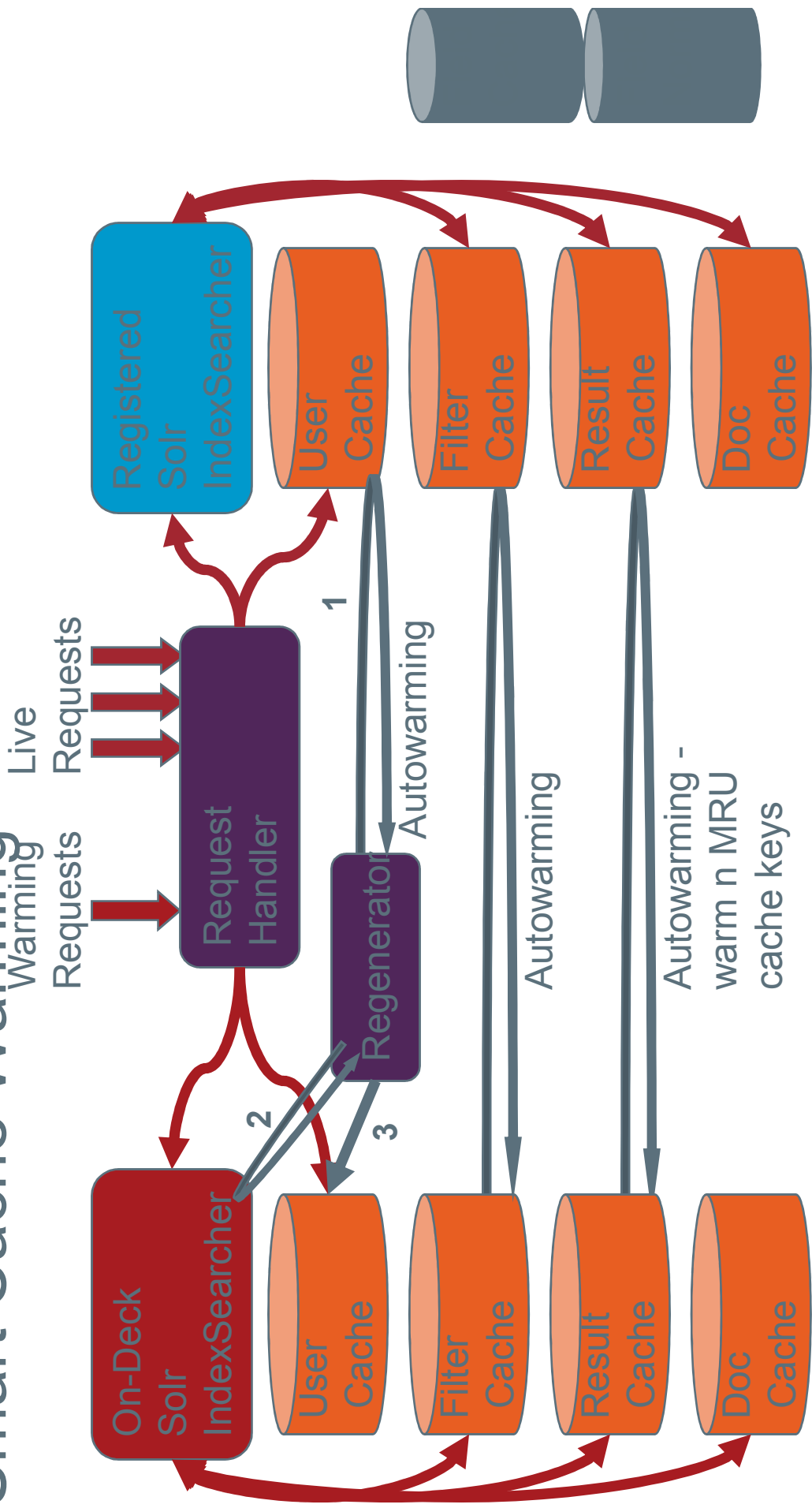


Warming

- » Lucene IndexReader warming
 - field norms
 - FieldCache
- » Cache warming
 - Configurable static requests to warm new Searchers
- » Smart Cache Warming (autowarming)
 - Using MRU items in the current cache to pre-populate the new cache



Smart Cache Warming



Schema

- » Lucene has no notion of a schema
 - Sorting - string vs. numeric
 - Ranges - val:42 included in val:[1 TO 5] ?
 - Lucene QueryParser has date-range support, but must guess.
- » Defines fields, their types, properties
- » Defines unique key field, default search field, Similarity implementation



Field Definitions

» Field Attributes: name, type, indexed, stored, multiValued, omitNorms

```
<field name="id" type="string" indexed="true" stored="true"/>
```

```
<field name="sku" type="textTight" indexed="true" stored="true"/>
```

```
<field name="name" type="text" indexed="true" stored="true"/>
```

```
<field name="reviews" type="text" indexed="true" stored="false"/>
```

```
<field name="category" type="text_ws" indexed="true" stored="true" multiValued="true"/>
```

» Dynamic Fields, in the spirit of Lucene!

```
<dynamicField name="*_i" type="sint" indexed="true" stored="true"/>
```

```
<dynamicField name="*_s" type="string" indexed="true" stored="true"/>
```

```
<dynamicField name="*_t" type="text" indexed="true" stored="true"/>
```



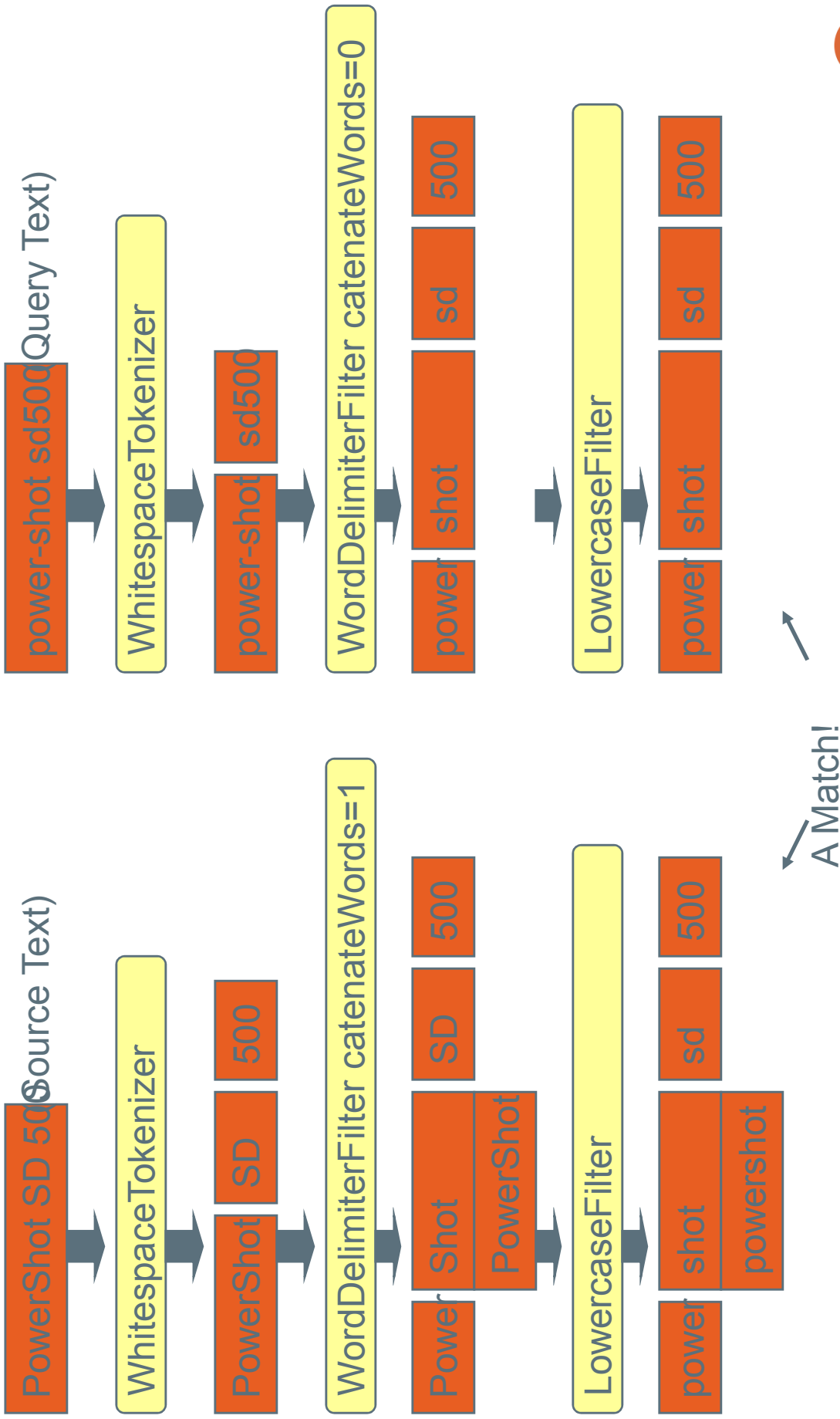
Field Type Definitions

```
<fieldtype name="sint" class="solr.SortableIntField" sortMissingLast="true"/>

<fieldtype name="text" class="solr.TextField">
  <analyzer>
    <tokenizer class="solr.StandardTokenizerFactory"/>
    <filter class="solr.LowerCaseFilterFactory"/>
    <filter class="solr.SynonymFilterFactory"
      synonyms="synonyms.txt"/>
    <filter class="solr.StopFilterFactory" words="stopwords.txt"/>
    <filter class="solr.EnglishPorterFilterFactory"
      protected="protwords.txt"/>
  </analyzer>
</fieldtype>
```



Analysis Example



Analysis Tool: Output

File Edit View History Bookmarks Tools Help

http://localhost:8983/solr/admin/analysis.jsp

Field name text

Field value (Index)
 verbose output
 highlight matches

The Quick/Brown Fox Jumped Over The Lazy Dog

Field value (Query)
 verbose output

brown fox?

Analyze

Index Analyzer

org.apache.solr.analysis.WhitespaceTokenizerFactory {}

term position	1	2	3	4	5	6	7	8
term text	The	Quick/Brown	Fox	Jumped	Over	The	Lazy	Dog
term type	word	word	word	word	word	word	word	word
source start,end	0,3	4,15	16,19	20,26	27,31	32,35	36,40	41,44
payload								

org.apache.solr.analysis.StopFilterFactory {enablePositionIncrements=true, words=stopwords.txt, ignoreCase=true}

term position	2	3	4	5	7	8
term text	Quick/Brown	Fox	Jumped	Over	Lazy	Dog
term type	word	word	word	word	word	word
source start,end	4,15	16,19	20,26	27,31	36,40	41,44
payload						

org.apache.solr.analysis.WordDelimiterFilterFactory {catenateWords=1, catenateNumbers=1, splitOnCaseChange=1, catenateAll=0, generateNumberParts=1, generateWordParts=1}

term position	2	3	4	5	6	8	9
term text	QuickBrown	Fox	Jumped	Over	Lazy	Dog	
term type	word	word	word	word	word	word	
source start,end	4,15	16,19	20,26	27,31	36,40	41,44	
payload							

Done

copyField

- » Analyze same field different ways
 - Boost exact-case or exact-punctuation matches
 - translations, thesaurus, soundex
- » Index multiple fields into single searchable field

```
<field name="title" type="text"/>
```

```
<field name="title_exact" type="text_ex" stored="false"/>
```

```
<field name="catchall" type="text" stored="false"/>
```

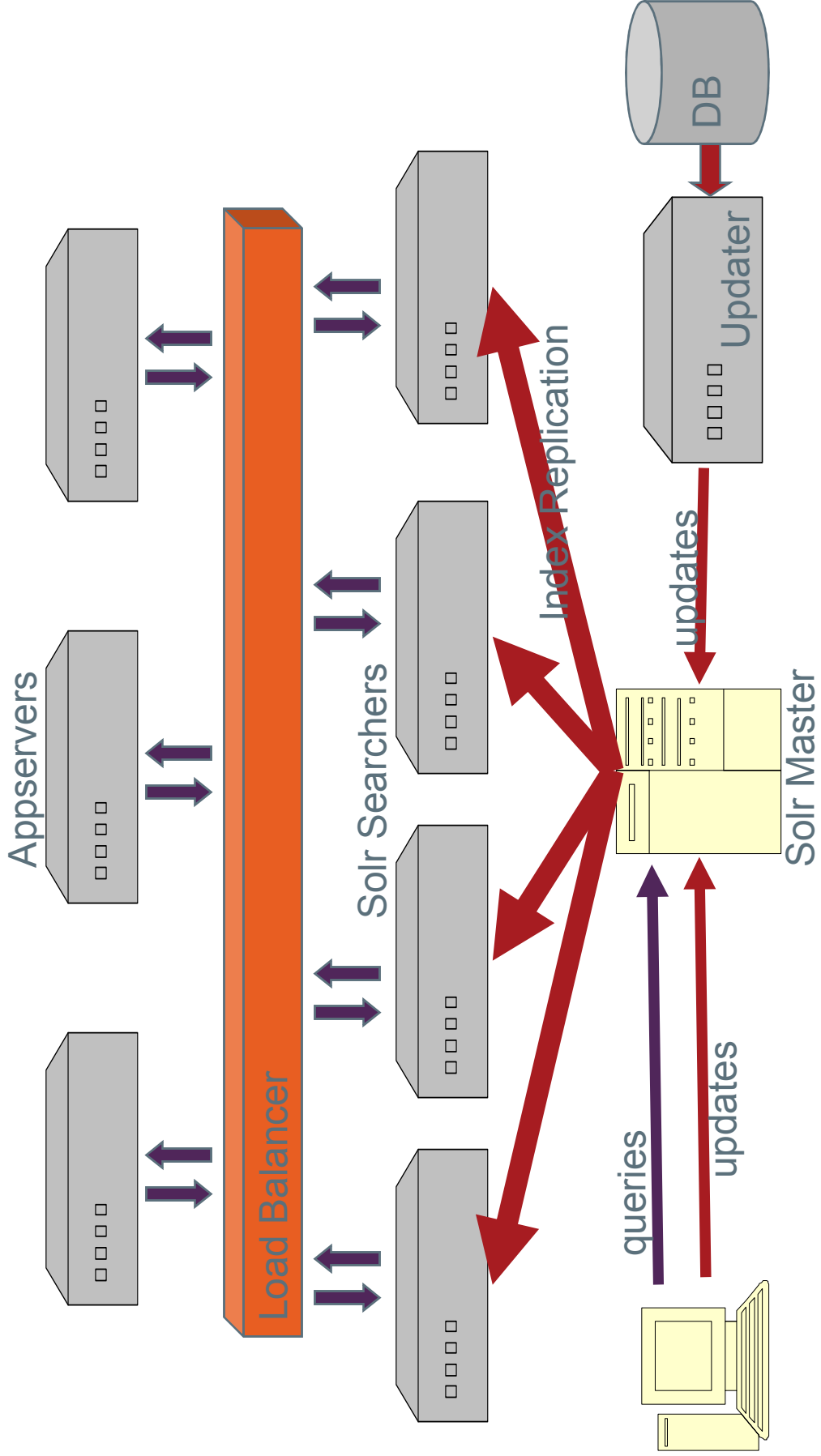
```
<copyField source="title" dest="text_exact"/>
```

```
<copyField source="title" dest="catchall"/>
```

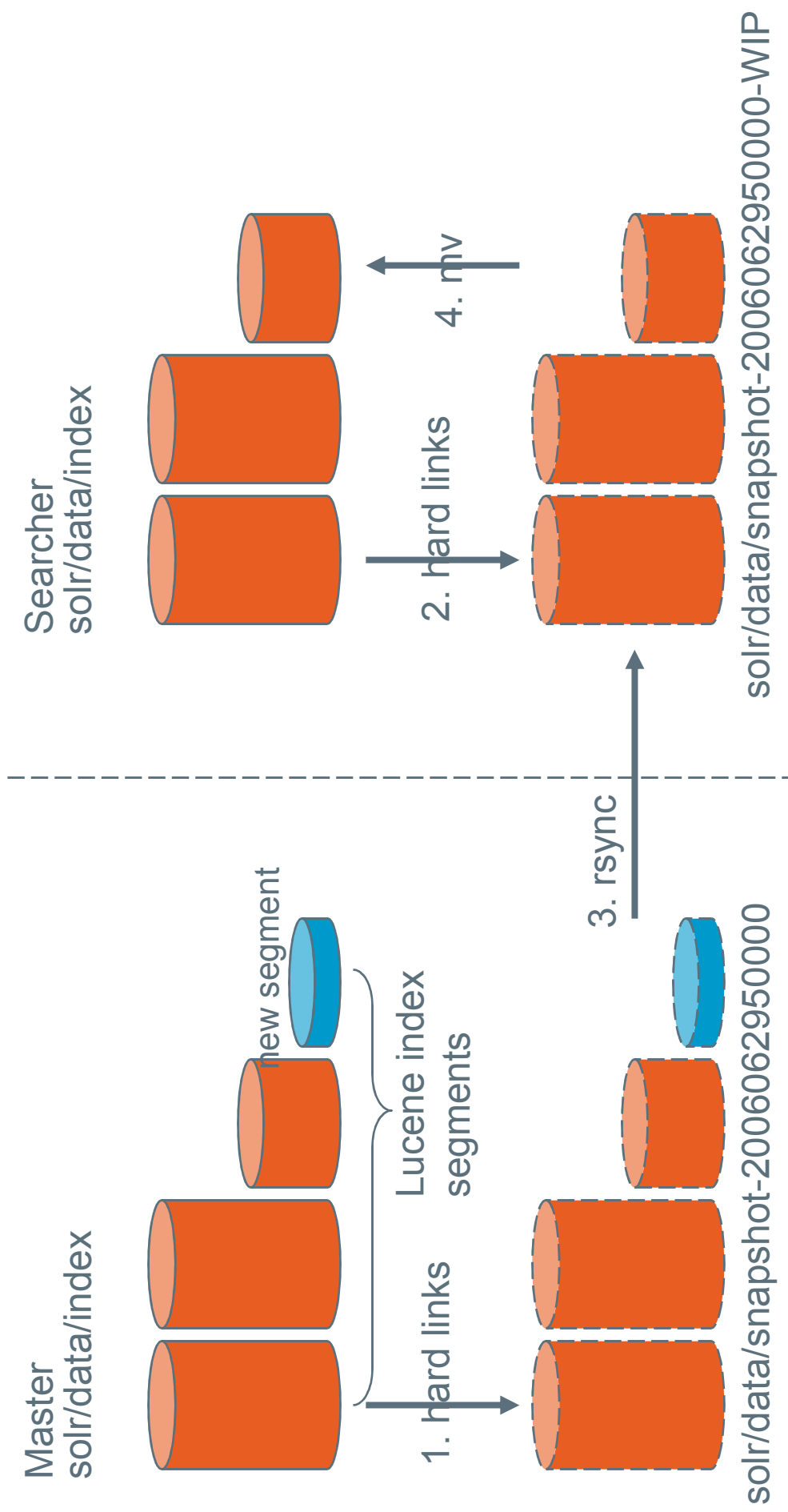
```
<copyField source="subject" dest="catchall"/>
```



High Availability



Replication

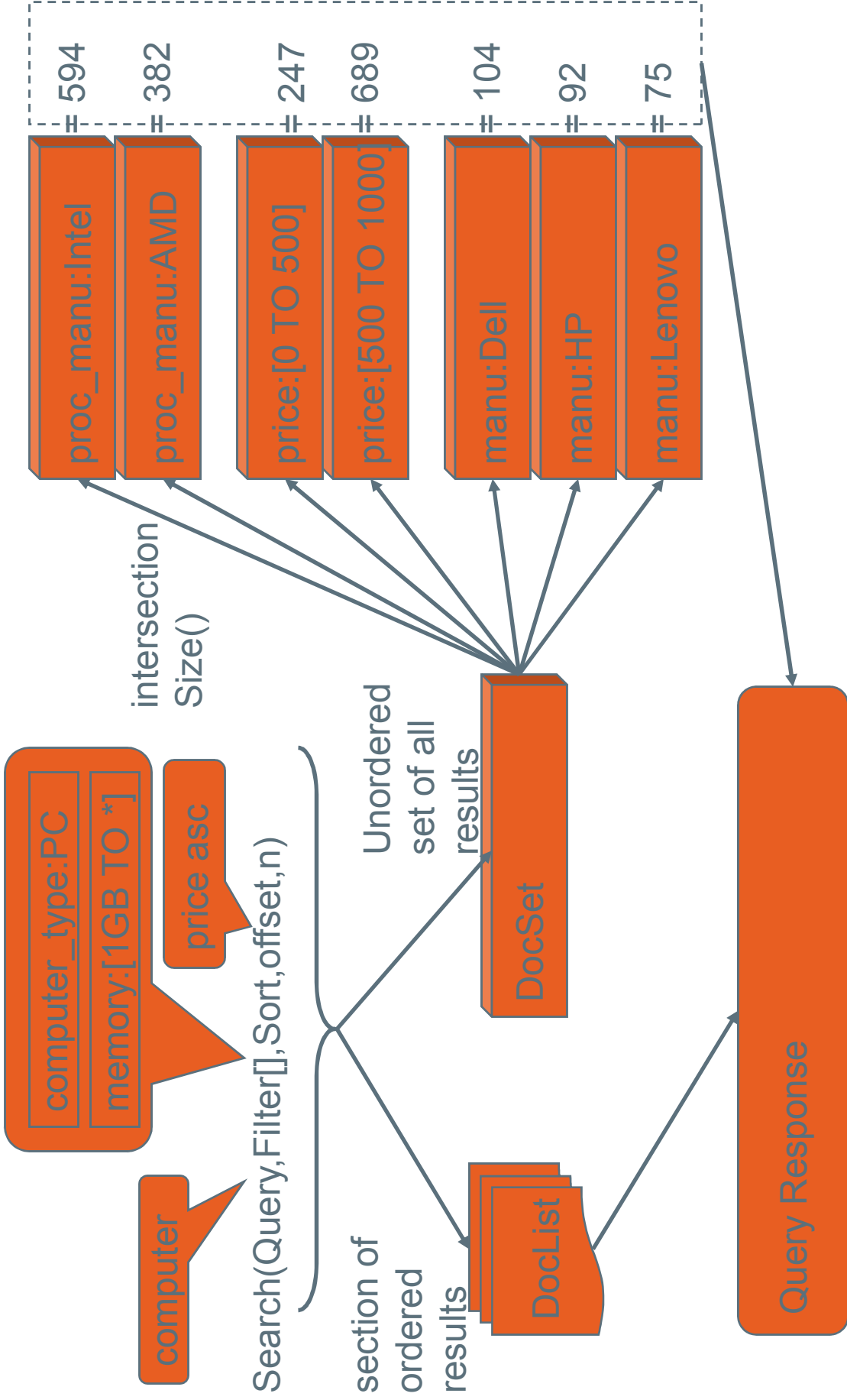


Replication

- » Replication scripts for efficiently mirroring an index on multiple machines.
 - snapshooter
 - snappuller
 - snapinstaller



Faceted Browsing



Querying Data: Facet Counts

- » Constraint counts can be computed for the whole result set using field values or explicit queries....

```
&facet=true&facet.field=cat&facet.field=inStock
```

```
&facet.query=price:[0 TO 10]&facet.query=price:[10 TO *]
```

```
...
```

```
<lst name="facet_counts">
```

```
<lst name="facet_queries">
```

```
<int name="price:[0 TO 10]">0</int>
```

```
<int name="price:[10 TO *]">13</int>
```

```
</lst>
```

```
<lst name="facet_fields">
```

```
<lst name="inStock">
```

```
<int name="true">10</int>
```

```
<int name="false">4</int>
```

```
</lst>
```

```
44
```

```
...
```



Querying Data: Facet Counts

File Edit View History Bookmarks Tools Help

http://siris-collections.si.edu/search/results.jsp?fq=type%3A%22Works+of+art%22&q=japanese+art&view

Smithsonian Institution
Research Information System

Cross Catalog Searching Center

search

Only return results with online media

home

list view grid view search history help

Search Results (308 documents - page 1 of 16) sort order: title | relevancy slideshow

search terms

- query:japanese art
- type: "Works of art"

browse options

browse & select

online media

- Images (118)
- Online collections (2)
- Finding aids (1)

type

- Works of art
 - Archival materials (216)
 - Books (35)
 - Exhibition catalogs (10)
 - Photographs (10)
 - Exhibitions (events) (9)
 - Pictorial works (6)
 - Diaries (3)
 - Biographies (2)
 - Catalogs (2)
- expand filter list

topic

- Art, Buddhist (14)
- Sculpture, Buddhist (14)
- Painting, Buddhist (9)

Done

Japanese Frets and Diapers
n.d. Photo-Li...

Two Scenes with Japanese Women Eating an...

Japanese Designs n.d.
Photo-Lithograph

Japanese Medallions n.d.
Photo-Lithograph...

Ethnographie n.d. Engraving

Carpenter's Tool n.d. Drawing

Carpenter's Tool n.d. Drawing

Carpenter's Tool n.d. Drawing

Carpenter's Tool n.d. Drawing

Web Admin Interface

- » Show Config, Schema, Distribution info
- » Query Interface
- » Statistics
 - Caches: lookups, hits, hitratio, inserts, evictions, size
 - RequestHandlers: requests, errors
 - UpdateHandler: adds, deletes, commits, optimizes
 - IndexReader, open-time, index-version, numDocs, maxDocs,
- » Analysis Debugger
 - Shows tokens after each Analyzer stage
 - Shows token matches for query vs index



The Admin Console

The screenshot shows the Solr Admin console in a web browser. The browser's address bar displays the URL `http://localhost:8983/solr/admin/`. The page header features the Solr logo and the text `coaster:8983` and `cwd=/home/hossman/tmp/solr1.3/apache-solr-1.3.0/example/SolrHome=solr/`. The main content area is divided into several sections:

- Solr**: A navigation menu with links for [\[SCHEMA\]](#), [\[CONFIG\]](#), [\[ANALYSIS\]](#), [\[SCHEMA BROWSER\]](#), [\[STATISTICS\]](#), [\[INFO\]](#), [\[DISTRIBUTION\]](#), [\[PING\]](#), [\[LOGGING\]](#), [\[JAVA PROPERTIES\]](#), and [\[THREAD DUMP\]](#).
- App server:** A section for application server management.
- Make a Query**: A section with a [\[FULL INTERFACE\]](#) link and a text input field containing the word `solr`. Below the input is a `Search` button.
- Assistance**: A section with links for [\[DOCUMENTATION\]](#), [\[ISSUE TRACKER\]](#), [\[SEND EMAIL\]](#), and [\[SOLR QUERY SYNTAX\]](#). It also displays the text `Current Time: Sun Sep 14 14:26:12 PDT 2008` and `Server Start At: Sun Sep 14 13:59:26 PDT 2008`.

The browser's status bar at the bottom shows the word `Done` and a small red icon with the letters `ABP`.



Odds and Ends

Querying Data: Results

» Canonical response format is XML...

```
<response>  
  <lst name="responseHeader">  
    <int name="status">0</int>  
    <int name="QTime">1</int>  
  </lst>  
  <result name="response" numFound="14" start="0">  
    <doc>  
      <arr name="cat">  
        <str>electronics</str>  
        <str>connector</str>  
      </arr>  
      <arr name="features">  
        <str>car power adapter, white</str>  
      </arr>  
      <str name="id">F8V7067-APL-KIT</str>
```



Querying Data: Highlighting

The screenshot shows a web browser window with the address bar containing the URL: `http://digg.com/search?s=solr&submit=Search§ion=all&type=all&area=all&sort=score`. The page title is "Search Digg". Below the title, there are navigation tabs for "All", "News", "Videos", "Images", and "Podcasts". A search bar contains the text "solr". To the right of the search bar, there are options for "All Stories" and "Sort Best Match First", along with a checkbox for "Include Buried Stories".

The search results are displayed as a list of items, each with a "digg it" button and a "Share" button. The first result is titled "Apache Solr 1.3 Released!" and has 1 digg. The second result is titled "Using Lucene Solr in Ruby for simplified faceted search" and has 3 diggs. The third result is titled "How to Index Data in Solr" and has 1 digg. The fourth result is titled "Installing and Configuring Solr on Gentoo Linux" and has 1 digg.

Each result includes a brief description and a "More..." link. For example, the first result states: "The Apache Solr team is happy to announce the availability of Solr 1.3.0 for public download. This version contains many enhancements and bug fixes, including Distributed Search, Multiple Index Support, Solr binary response, Search Components, DataImportHandler, and more." The second result states: "Lucene Solr is a powerful free text search engine that also offers faceted queries, but they can be a bit messy and cryptic. DeSolr is a new ruby gem that makes Solr faceting and filtering simpler and a little more rubyish." The third result states: "The exampledocs directory contains samples of the types of instructions Solr expects, as well as a java utility for posting them from the command line (a post.sh shell script is also available." The fourth result states: "Lucene is a very nice, robust and flexible search engine. The ways you can benefit from Lucene varies from small library system to a large e-commerce portal. To unleash Lucene, you either have to use Lucene ported to your favorite language or install Solr and let Solr handle communication with Lucene."

Querying Data: Highlighting

- » Generates summary "fragments" of stored fields showing matches....

```
&hl=true&hl.fl=features&hl.fragsize=30
```

```
...
```

```
<lst name="highlighting">
```

```
<lst name="F8V7067-APL-KIT">
```

```
<arr name="features">
```

```
<str>car power &lt;em&gt;adapter&lt;/em&gt;, white</str>
```

```
</arr>
```

```
</lst>
```

```
...
```



Describing Your Data

- » schema.xml is where you configure the options for various fields.
- Is it a number? A string? A date?
- Is there a default value for documents that don't have one?
- Is it created by combining the values of other fields?
- Is it stored for retrieval?
- Is it indexed? If so is it parsed? If so how?
- Is it a unique identifier?



Fields

- `<field>` Describes How You Deal With Specific Named Fields
- `<dynamicField>` Describes How To Deal With Fields That Match A Glob (Unless There Is A Specific `<field>` For Them)
- `<copyField>` Describes How To Construct Fields From Other Fields

```
<field name="title" type="text" stored="false" />
```

```
<dynamicField name="price*" type="sfloat" indexed="true" />
```

```
<copyField source="*" dest="catchall" />
```



Field Types

- Every Field Is Based On A `<fieldType>` Which Specifies:
 - The Underlying Storage Class (`FieldType`)
 - The Analyzer To Use Or Parsing If It Is A Text Field
- OOTB Solr Has 18 `FieldType` Classes

```
<fieldType name="sfloat" class="solr.SortableFloatField"
```

```
  sortMissingLast="true" omitNorms="true" />
```

```
<fieldType name="string" class="solr.StrField"
```

```
  indexed="true" stored="true" />
```

```
<fieldType name="unstored" class="solr.StrField"
```

```
  indexed="true" stored="false" />
```



Analyzers

- 'Analyzer' Is A Core Lucene Class For Parsing Text
- Solr Includes 18 Lucene Analyzers That Can Be Used OOTB If They Meet Your Needs

```
<fieldType name="text_greek" class="solr.TextField">  
  <analyzer class="org.apache.lucene.analysis.el.GreekAnalyzer"/>  
</fieldType>
```

...BUT WAIT!



Tokenizers And TokenFilters

- Analyzers Are Typical Comprised Of Tokenizers And TokenFilters
 - Tokenizer: Controls How Your Text Is Tokenized
 - TokenFilter: Mutates And Manipulates The Stream Of Tokens
- Solr Lets You Mix And Match Tokenizers and TokenFilters In Your schema.xml To Define Analyzers On The Fly
- OOTB Solr Has Factories For 12 Tokenizers and 36 TokenFilters
- Many Factories Have Customization Options -- Limitless Combinations



Tokenizers And TokenFilters

```
<fieldType name="text" class="solr.TextField">
  <analyzer type="index">
    <tokenizer class="solr.WhitespaceTokenizerFactory"/>
    <filter class="solr.StopFilterFactory words="stopwords.txt"/>
    <filter class="solr.WordDelimiterFilterFactory"
      generateWordParts="1" generateNumberParts="1"/>
    <filter class="solr.LowerCaseFilterFactory"/>
    <filter class="solr.EnglishPorterFilterFactory"
      protected="protwords.txt"/>
  </analyzer>
  <analyzer type="query">
    <tokenizer class="solr.WhitespaceTokenizerFactory"/>
    <filter class="solr.SynonymFilterFactory"
      synonyms="synonyms.txt" expand="true"/>
  </analyzer>
</fieldType>
```

...



Notable Tokenizers|Filters)

- StandardTokenizerFactory
- HTMLStripWhitespaceTokenizerFactory
- KeywordTokenizerFactory
- NGramTokenizerFactory
- PatternTokenizerFactory

- EnglishPorterFilterFactory
- SynonymFilterFactory
- StopFilterFactory
- ISOLatin1AccentFilterFactory
- PatternReplaceFilterFactory



Interacting With Your Data

- » `solrconfig.xml` is where you configure options for how this Solr instance should behave.
- »
 - Low-Level Index Settings
 - Performance Settings (Cache Sizes, etc...)
 - Types of Updates Allowed
 - Types of Queries Allowed

Note:

- `solrconfig.xml` depends on `schema.xml`.
- `schema.xml` does not depend on `solrconfig.xml`.



Request Handlers

- Type Of Request Handler Determines Options, Syntax, And Logic For Processing Requests
- OOTB Indexing Handlers:
 - XmlUpdateRequestHandler
 - CSVRequestHandler
 - DataImportHandler
- OOTB Searching Handler:
 - SearchHandler + QParsers



Example: Handler Configuration

```
<requestHandler name="/select" class="solr.SearchHandler" />
<requestHandler name="/simple" class="solr.SearchHandler" >
  <lst name="defaults">
    <str name="defType">dismax</str>
    <str name="qf">catchall</str>    </lst>
</requestHandler>
<requestHandler name="/complex" class="solr.SearchHandler" >
  <lst name="defaults">
    <str name="defType">dismax</str>
    <str name="qf">features^1 name^2</str>  </lst>
    <lst name="appends">
      <str name="fq">inStock:true</str>    </lst>
    <lst name="invariants">
      <str name="facet">>false</str>
    ...
```



Output: Response Writers

- Response Format Can Be Controlled Independently From Request Handler Logic
- Many Useful Response Writers OOTB

`http://solr/select?q=electronics`

`http://solr/select?q=electronics&wt=xml`

`http://solr/select?q=electronics&wt=json`

`http://solr/select?q=electronics&wt=python`

`http://solr/select?q=electronics&wt=ruby`

`http://solr/select?q=electronics&wt=php`

`http://solr/select?q=electronics&wt=xslt&tr=example.xsl`

62 `<queryResponseWriter name="xml" default="true"`



Installing Solr

- Put The solr.war Where Your Favorite Servlet Container Can Find It
- Create A "Solr Home" Directory
- Steal The Example solr/conf Files
- Point At Your Solr Home Using Either:
 - JNDI
 - System Properties
 - The Current Working Directory

(Or just use the Jetty example setup.)

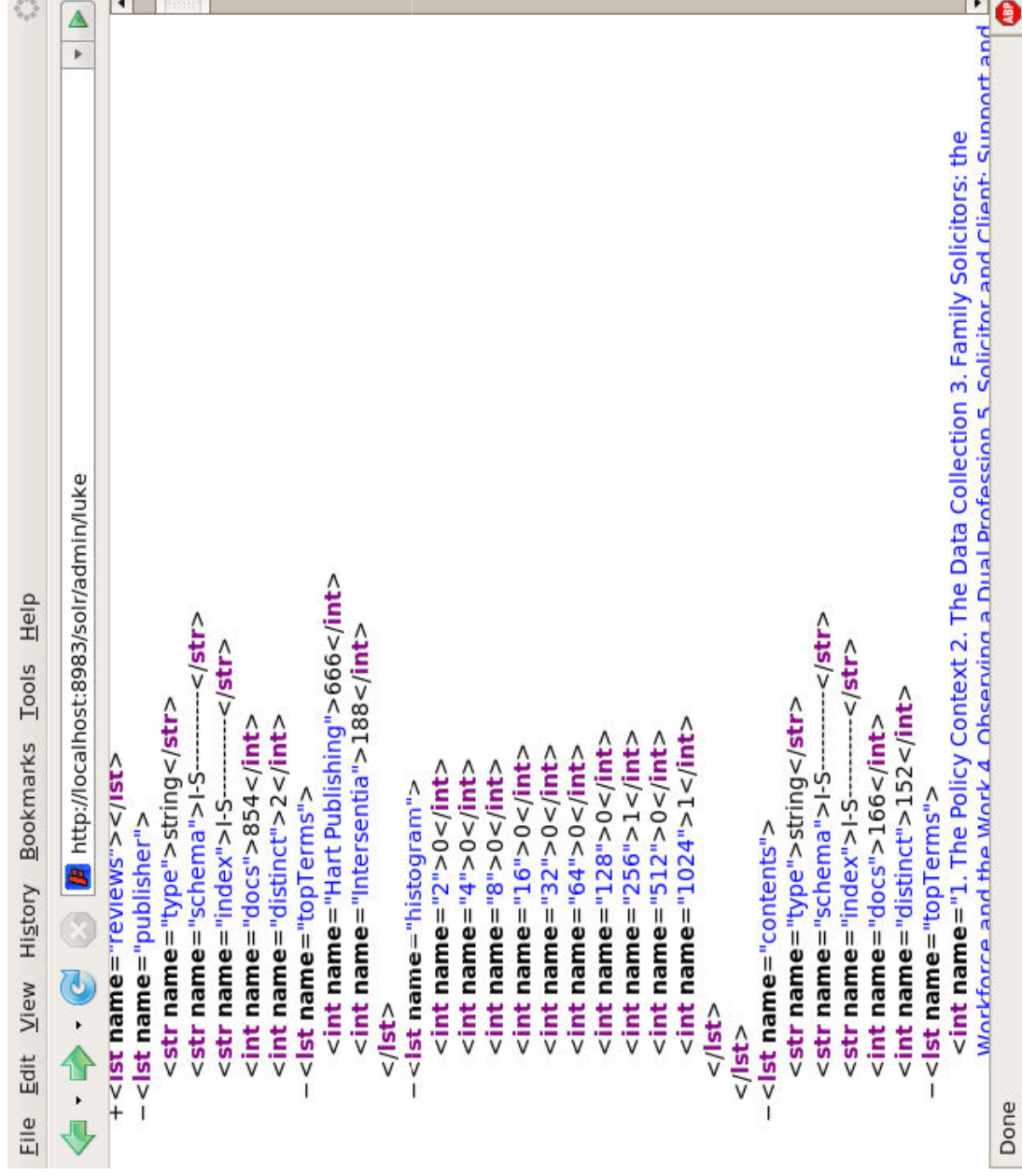


Understanding The Data: Luke

- The LukeRequestHandler Is Based On A Popular Lucene GUI App For Debugging Indexes (Luke)
- Allows Introspection Of Field Information:
 - Options From The Schema (Either Explicit Or Inherited From Field Type)
 - Statistics On Unique Terms And Terms With High Doc Frequency
 - Histogram Of Terms With Doc Frequency Above Set Thresholds
- Helpful In Understanding The Nature Of Your Data
- Schema Browser: Luke On Steroids



Example: Luke Output



The screenshot shows a web browser window with the address bar containing `http://localhost:8983/solr/admin/luke`. The main content area displays XML output for the 'luke' tool. The output is a list of XML elements, each representing a different field in the index. The fields and their values are:

- `<str name="reviews"></str>`
- `<lst name="publisher">`
 - `<str name="type">string</str>`
 - `<str name="schema">l-S-----</str>`
 - `<str name="index">l-S-----</str>`
 - `<int name="docs">854</int>`
 - `<int name="distinct">2</int>`
- `<lst name="topTerms">`
 - `<int name="Hart Publishing">666</int>`
 - `<int name="Intersentia">188</int>`
- `</lst>`
- `<lst name="histogram">`
 - `<int name="2">0</int>`
 - `<int name="4">0</int>`
 - `<int name="8">0</int>`
 - `<int name="16">0</int>`
 - `<int name="32">0</int>`
 - `<int name="64">0</int>`
 - `<int name="128">0</int>`
 - `<int name="256">1</int>`
 - `<int name="512">0</int>`
 - `<int name="1024">1</int>`
- `</lst>`
- `<lst name="contents">`
 - `<str name="type">string</str>`
 - `<str name="schema">l-S-----</str>`
 - `<str name="index">l-S-----</str>`
 - `<int name="docs">166</int>`
 - `<int name="distinct">152</int>`
- `<lst name="topTerms">`
 - `<int name="1. The Policy Context 2. The Data Collection 3. Family Solicitors: the Workforce and the Work 4. Observing a Dual Profession 5. Solicitor and Client- Summary and`

The browser's status bar at the bottom shows the word "Done".

Example: Schema Browser

File Edit View History Bookmarks Tools Help

http://localhost:8080/books-solr/admin/schema.jsp

Schema: Indexed, Stored, Omit Norms, Sort Missing Last

Index: Indexed, Stored, Omit Norms

Copied Into: [CATCHALL](#)

Index Analyzer: org.apache.solr.schema.FieldType\$DefaultAnalyzer

Query Analyzer: org.apache.solr.schema.FieldType\$DefaultAnalyzer

Docs: 814

Distinct: 51

Top 10 Terms

term	frequency
LAW051000	174
LAW052000	85
LAW013000	67
LAW005000	44
LAW021000	32
LAW018000	30
LAW001000	24
LAW026000	22
LAW054000	21
LAW014010	20

Histogram

Frequency Bin	Number of Terms
2	2
4	8
8	10
16	7
32	1
64	2
128	1
256	1

Done

Refining Your Schema

- Pick Field Types That Make Sense
- Pick Analyzers That Make Sense
- Use <copyField> To Make Multiple Copies Of Fields For Different Purposes:
 - Faceting
 - Sorting
 - Loose Matching
 - Etc...



Example: "BIC" Codes

- » <!-- used by the bic field, a prefix based code -->
- » <fieldType name="bicgram" class="solr.TextField" >
- » <analyzer type="index">
- » <tokenizer class="solr.EdgeNGramTokenizerFactory"
- » minGramSize="1"
- » maxGramSize="100"
- » side="front" />
- » <filter class="solr.LowerCaseFilterFactory"/>
- » </analyzer>
- » <analyzer type="query">
- » <tokenizer class="solr.WhitespaceTokenizerFactory" />
- » <filter class="solr.LowerCaseFilterFactory"/>



Search Components

- Default Components That Power SearchHandler
 - QueryComponent
 - HighlightComponent
 - FacetComponent
 - MoreLikeThisComponent
 - DebugComponent
- Additional Components You Can Configure
 - SpellCheckComponent
 - QueryElevationComponent



Score Explanations

- Why Did Document X Score Higher Than Y?
- Why Didn't Document Z Match At All?
- Debugging Options Can Answer Both Questions....
 - idf - How Common A Term Is In The Whole Index
 - tf - How Common A Term Is In This Document
 - fieldNorm - How Significant Is This Field In This Document (Usually Based On Length)
 - boost - How Important The Client Said This Clause Is
 - coordFactor - How Many Clauses Matched

`&debugQuery=true&explainOther=documentId:Z`



Example: Score Explanations

- » `<str name="id=9781841135779,internal_docid=111">`
- » **0.30328625** = (MATCH) fieldWeight(catchall:law in 111), product of:
- » **3.8729835** = $\text{tf}(\text{termFreq}(\text{catchall:law})=\mathbf{15})$
- » $1.0023446 = \text{idf}(\text{docFreq}=851)$
- » **0.078125** = $\text{fieldNorm}(\text{field}=\text{catchall}, \text{doc}=111)$
- » `</str>`
- » ...
- » `<str name="id=9781841135335,internal_docid=696">`
- » **0.26578674** = (MATCH) fieldWeight(catchall:law in 696), product of:
- » **4.2426405** = $\text{tf}(\text{termFreq}(\text{catchall:law})=\mathbf{18})$
- » $1.0023446 = \text{idf}(\text{docFreq}=851)$
- » **0.0625** = $\text{fieldNorm}(\text{field}=\text{catchall}, \text{doc}=696)$
- » `</str>`



» DataImportHandler

Builds and incrementally updates indexes based on configured SQL or XPath queries.

```
<entity name="item" pk="ID" query="select * from ITEM"
deltaQuery="select ID ... where
    ITEMDATE > '${dataimporter.last_index_time}'">
<field column="NAME" name="name" />
...
<entity name="f" pk="ITEMID"
query="select DESC from FEATURE where ITEMID='${item.ID}'"
deltaQuery="select ITEMID from FEATURE where
    UPDATEDATE > '${dataimporter.last_index_time}'"
parentDeltaQuery="select ID from ITEM where ID=${f.ITEMID}">
<field name="features" column="DESC" />
```





References

Resources

- » Home Page
 - <http://lucene.apache.org/solr>
 - Tutorial
 - <http://wiki.apache.org/solr/>
- » Mailing Lists
 - solr-user-subscribe@lucene.apache.org
 - solr-dev-subscribe@lucene.apache.org

