How SOLR Search Works

Avesh Srivastava – 26th May, 2015

Introduction

What is Solr?

- Solr is an open-source search platform which is used to build search applications.
- It was built on top of Lucene (full text search engine).
- Solr is enterprise-ready, fast and highly scalable.
- The applications built using Solr are sophisticated and deliver high performance.
- Solr is a ready to deploy, search/storage engine optimized to search large volumes of text-centric data.

History of Solr

It was Yonik Seely who created Solr in 2004 in order to add search capabilities to the company website of CNET Networks. In Jan 2006, it was made an open-source project under Apache Software Foundation. Its latest version, Solr 6.0, was released in 2016 with support for execution of parallel SQL queries.

Solr is the second-most popular enterprise search engine after Elasticsearch

Features of Apache Solr

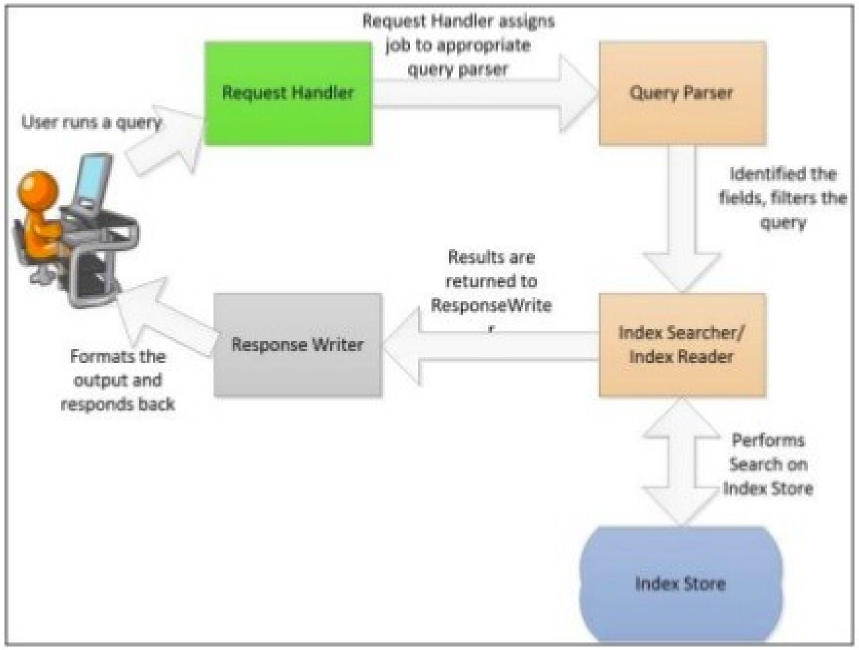
- **Restful APIs** − To communicate with Solr, it is not mandatory to have Java programming skills. Instead you can use restful services to communicate with it. We enter documents in Solr in file formats like XML, JSON and .CSV and get results in the same file formats.
- **Full text search** − Solr provides all the capabilities needed for a full text search such as tokens, phrases, spell check, wildcard, and auto-complete.
- **Enterprise ready** − According to the need of the organization, Solr can be deployed in any kind of systems (big or small) such as standalone, distributed, cloud, etc.

- **Flexible and Extensible** − By extending the Java classes and configuring accordingly, we can customize the components of Solr easily.
- **NoSQL database** − Solr can also be used as big data scale NOSQL database where we can distribute the search tasks along a cluster.
- **Admin Interface** − Solr provides an easy-to-use, user friendly, feature powered, user interface, using which we can perform all the possible tasks such as manage logs, add, delete, update and search documents.
- **Highly Scalable** − While using Solr with Hadoop, we can scale its capacity by adding replicas.
- **Text-Centric and Sorted by Relevance** − Solr is mostly used to search text documents and the results are delivered according to the relevance with the user’s query in order.

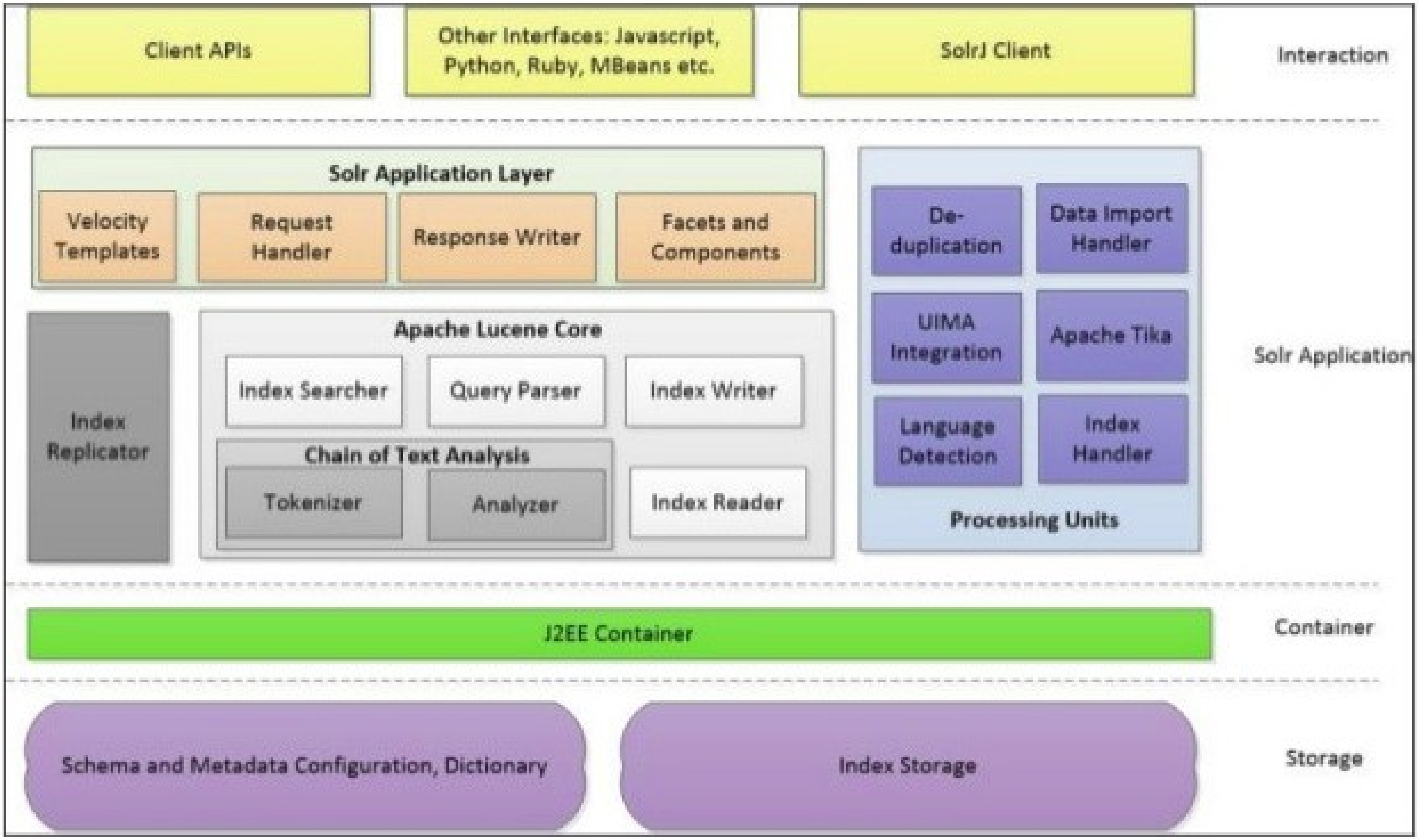
How Solr Works?

SOLR Architecture – Searching

- User enters query
- Parse the query, i.e. apply the required filters and tokenizers
- Converted to tokens
- Parallel search across multiple indexes (per field)
- Score all the documents
- Sort in async fashion


Request Handler assigns job to appropriate query parser
Request Handler
Query Parser
User runs a query
Identified the fields, filters the query
Results are returned to ResponseWriter
Response Writer
Index Searcher/ Index Reader
Formats the output and responds back
Performs Search on Index Store
Index Store

Solr Architecture



Solr Usage

Some common usages of solr are

- Basic Search
- Advanced full-text searching
- Geospatial searching
- Auto Complete
- Filters
- Sorting
- Facets & More

Indexing In Solr

Types of Index Updates

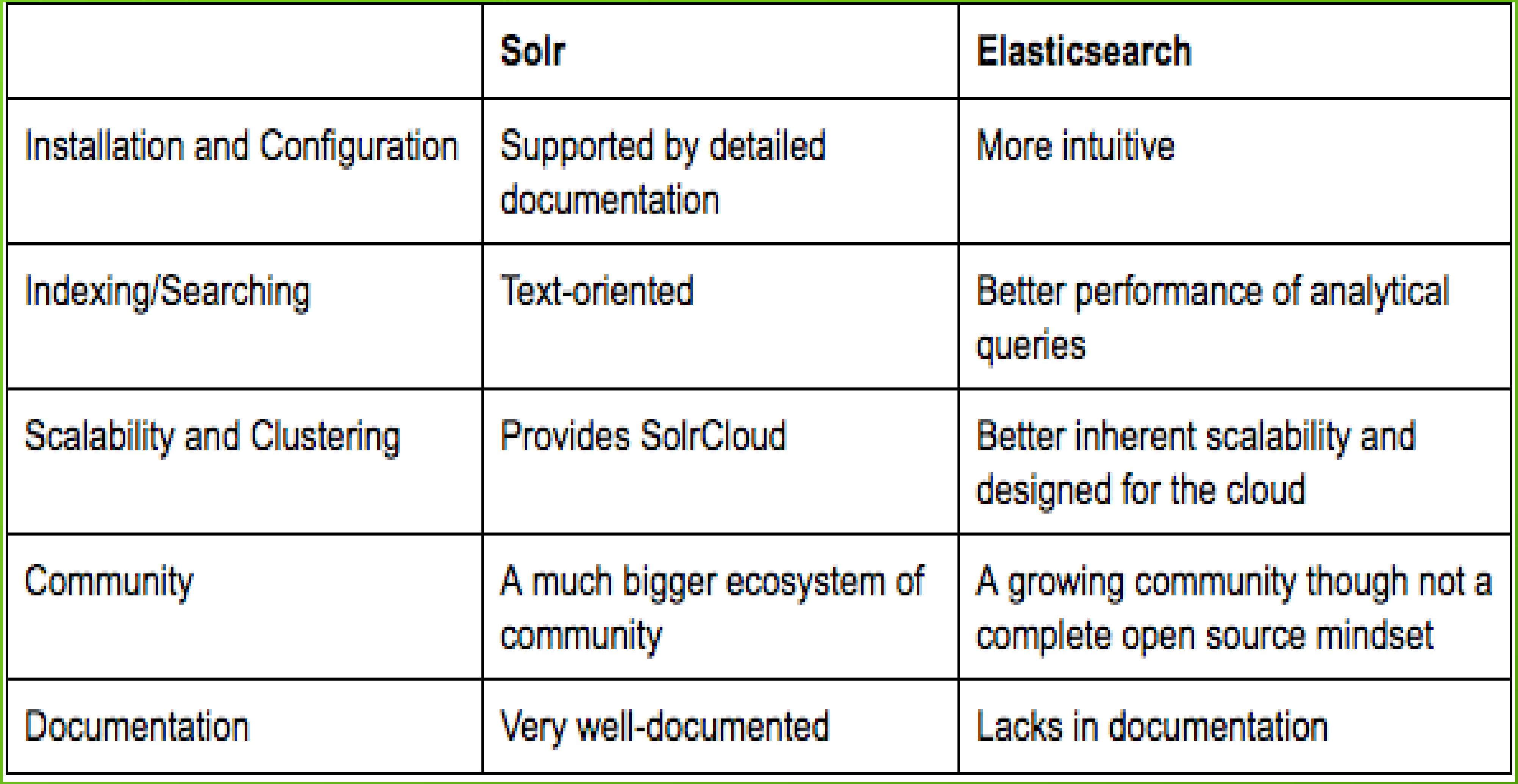
- Instant Index
- Incremental Indexing
- Full Indexing

Index Update Strategies

- Instant/Incremental Index cannot happen continuously
- To much causes performance degradation
- Full Index periodically to optimize the index

Other Search Technologies

ElasticSearch | Splunk | Sphinx



	Solr	**Elasticsearch**
Installation and Configuration	Supported by detailed documentation	More intuitive
Indexing/Searching	Text-oriented	Better performance of analytical queries
Scalability and Clustering	Provides SolrCloud	Better inherent scalability and designed for the cloud
Community	A much bigger ecosystem of community	A growing community though not a complete open source mindset
Documentation	Very well-documented	Lacks in documentation

Installation & Configuration

Solr Installation

Update your System

To update your system, execute the following command to update your system with latest patches and updates.

```
sudo apt-get update && apt-get upgrade -y
```

Install Java JDK

```
sudo apt-get -y install openjdk-7-jdk ( For Ubuntu )

sudo yum install java-1.7.0-openjdk-devel ( For CentOS )

sudo mkdir /usr/java

sudo ln -s /usr/lib/jvm/java-7-openjdk-amd64     /usr/java/default
```

Solr Installation

Download and Install Solr

We can now start the real installation of Solr. First, download all files and uncompress them:

```
cd /opt

sudo wget http://archive.apache.org/dist/lucene/solr/4.7.2/solr-4.7.2.tgz

sudo tar -xvf solr-4.7.2.tgz

sudo mkdir solr

sudo cp -R solr-4.7.2/example/* solr-4.7.2/contrib solr-4.7.2/dist  /opt/solr

cd /opt/solr

sudo java -jar start.jar
```

Check if it works by visiting http://localhost:8983/solr

Solr Installation

- Now your solr directory contain an instance of the Jetty Servlet container setup to run SOLR using an example configuration
- Prefix nohup to prevent the SOLR from being aborted if you log out or exit the shell as it runs into the background process.

First create a process output log file.

```
sudo nano output.log
chmod 777 output.log
```

and now start solr server with nohup

```
sudo nohup java -jar start.jar > output.log 2>&1 &
```

Overview of Documents, Fields, and Schema Design

Indexing : The part where you feed in all the information is called indexing or updating.

Document: Solr's basic unit of information is a document, which is a set of data that describes something.Documents are composed of fields, which are more specific pieces of information.Eg: A document about a person could contain their name, age, address, gender, and so on.

Fields: The definition of fields is flexible, it could be int, character, floating point etc. Eg. Person name is text field (character data), Age is int field and product price field might be a floating point number so that it could contain decimal values like 150.75 , 22.50 etc.

Overview of Documents, Fields, and Schema Design

It is mandatory to specify the field type, so that solr can understand what kind of data a filed should contain. When you add a document, Solr takes the information in the document's fields and adds that information to an index. When you perform a query, Solr can quickly consult the index and return the matching documents.

```
<field name="price" type="float" default="0.0"
 indexed="true" stored="true"/>
```

Field Types which are commonly used:

*** String**	*** text_general**
*** Float**	*** Int**
*** Int**	*** Date**

Overview of Documents, Fields, and Schema Design

Fields Default Properties

Property	Description	Value	Implicit Default
Indexed	If true, the value of the field can be used in queries to retrieve matching documents.	True / False	True
Large	Large fields are always lazy loaded and will only take up space in the document cache if the actual value is < 512KB. This option requires stored="true" and multiValued="false". It's intended for fields that might have very large values so that they don't get cached in memory.	True / False	False
Multi Value	If true, indicates that a single document might contain multiple values for this field type.	True / False	False

Overview of Documents, Fields, and Schema Design

Fields Default Properties

Property	Description	Value	Implicit Default
omitNorms	If true, omits the norms associated with this field (this disables length normalization for the field, and saves some memory). Defaults to true for all primitive (non-analyzed) field types, such as int, float, data, bool, and string. Only full-text fields or fields need norms.	True / False	*
required	Instructs Solr to reject any attempts to add a document which does not have a value for this field. This property defaults to false.	True / False	False
stored	If true, the actual value of the field can be retrieved by queries.	True / False	True

Solr Configuration

Load test data

To add documents to the index, use the post.jar (or post.sh script) in the solr/exampledocs subdirectory (while Solr is running), for example:

```
-> cd exampledocs
-> java -jar post.jar *.xml
-> Or:  sh post.sh *.xml
```

Solr Configuration

Create New Collection

Make a copy of given sample collection folder (Eg: Collection1)

```
-> sudo cp -R /opt/solr/solr/collection1 /opt/solr/solr/mycollection
```

Go into your newly created directory and change the collection name given in core.properties file.

```
-> cd /opt/solr/solr/mycollection/
-> sudo nano core.properties (Eg: name=mycollection)
```

- Download mysql JDBC connector from

  http://www.java2s.com/Code/Jar/c/Downloadcommysqljdbc515jar.htm

- Create a folder “db” under /opt/solr/contrib/ folder and then copy the downloaded file into it.

Solr Configuration

Include dependencies

Now go into the config folder and add the following lines at the beginning of solrconfig.xml file (At Ln no 75)

```
<lib dir="${solr.install.dir:../..}/dist/" regex="solr-dataimporthandler-.*\.jar" />

<lib dir="${solr.install.dir:../..}/contrib/db" regex="com.mysql.jdbc_\d.*\.jar" />
```

Now add below lines under the request handler section (At Ln no 830)

```
<requestHandler name="/dataimport" class="solr.DataImportHandler">
       <lst name="defaults">
         <str name="config">db-data-config.xml</str>
       </lst>
  </requestHandler>
```

Solr Configuration

Include dependencies

Open schema.xml file

```
sudo nano /opt/solr/solr/mycollection/conf/db-data-config.xml
```

Remove all un-neccessary fields and add new fields as per your requirement under the <field> tag (after ln no. 123)

Now create a file db-data-config.xml under conf directory

```
sudo nano /opt/solr/solr/mycollection/conf/db-data-config.xml
```

In this file we will write our data import queries.

Restart solr and that's it. Your new solr instance is configured..

The End.