







# Apache Solr vs Elasticsearch

## The Feature Smackdown






### API

Feature	Solr 6.2.1	ElasticSearch 5.0
Format	XML, CSV, JSON	JSON
HTTP REST API	✓	✓
Binary API 	✓ SolrJ	✓ TransportClient, Thrift (through a <a href="#">plugin</a> )
JMX support	✓	✗ ES specific stats are exposed through the REST API
Official client libraries 	Java	Java, Groovy, PHP, Ruby, Perl, Python, .NET, Javascript <a href="#">Official list of clients</a>
Community client libraries 	PHP, Ruby, Perl, Scala, Python, .NET, Javascript, Go, Erlang, Clojure	Clojure, Cold Fusion, Erlang, Go, Groovy, Haskell, Java, JavaScript, .NET, OCaml, Perl, PHP, Python, R, Ruby, Scala, Smalltalk, Vert.x <a href="#">Complete list</a>
3rd-party product integration (open-source) 	Drupal, Magento, Django, ColdFusion, Wordpress, OpenCMS, Plone, Typo3, ez Publish, Symfony2, Riak (via Yokozuna)	Drupal, Django, Symfony2, Wordpress, CouchBase
3rd-party product integration (commercial) 	DataStax Enterprise Search, Cloudera Search, Hortonworks Data Platform, MapR	SearchBlox, Hortonworks Data Platform, MapR etc <a href="#">Complete list</a>
Output 	JSON, XML, PHP, Python, Ruby, CSV, Velocity, XSLT, native Java	JSON, XML/HTML (via <a href="#">plugin</a> )

## Infrastructure


Feature	Solr 6.2.1	ElasticSearch 5.0
Master-slave replication	✓ <b>Only in non-SolrCloud.</b> In SolrCloud, behaves identically to ES.	✗ Not an issue because shards are replicated across nodes.
Integrated snapshot and restore	Filesystem	Filesystem, AWS Cloud Plugin for S3 repositories, HDFS Plugin for Hadoop environments, Azure Cloud Plugin for Azure storage repositories




## Indexing









Feature	Solr 6.2.1	ElasticSearch 5.0
Data Import	DataImportHandler - JDBC, CSV, XML, Tika, URL, Flat File	<b>[DEPRECATED in 2.x]</b> Rivers modules - ActiveMQ, Amazon SQS, CouchDB, Dropbox, DynamoDB, FileSystem, Git, GitHub, Hazelcast, JDBC, JMS, Kafka, LDAP, MongoDB, neo4j, OAI, RabbitMQ, Redis, RSS, Sofa, Solr, St9, Subversion, Twitter, Wikipedia
ID field for updates and deduplication	✓	✓
DocValues 	✓	✓
Partial Doc Updates 	✓ with stored fields	✓ with <code>_source</code> field
Custom Analyzers and Tokenizers 	✓	✓
Per-field analyzer chain 	✓	✓
Per-doc/query analyzer chain 	✗	✓

Feature	Solr 6.2.1	ElasticSearch 5.0
Index-time synonyms 	✓	✓ Supports Solr and Wordnet synonym format
Query-time synonyms 	✓ especially via <a href="#">hon-lucene-synonyms</a>	✗ Technically, yes, but practically no because multi-word/phrase query-time synonyms are not supported. See <a href="#">ES docs</a> and <a href="#">hon-lucene-synonyms</a> blog for nuances.
Multiple indexes 	✓	✓
Near-Realtime Search/Indexing 	✓	✓
Complex documents 	✓	✓
Schemaless 	✓ 4.4+	✓
Multiple document types per schema 	✗ One set of fields per schema, one schema per core	✓
Online schema changes 	✓ Schemaless mode or via dynamic fields.	✓ Only backward-compatible changes.
Apache Tika integration 	✓	✓
Dynamic fields 	✓	✓
Field copying 	✓	✓ via multi-fields
Hash-based deduplication 	✓	✓ <a href="#">Murmur plugin</a> or <a href="#">ER plugin</a>



## Searching






Feature	Solr 6.2.1	ElasticSearch 5.0
Lucene Query parsing 	✓	✓

Feature	Solr 6.2.1	ElasticSearch 5.0
Structured Query DSL 	✗ Need to programmatically create queries if going beyond Lucene query syntax.	✓
Span queries 	✓ via <a href="#">SOLR-2703</a>	✓
Spatial/geo search 	✓	✓
Multi-point spatial search 	✓	✓
Faceting 	✓	✓ Top N term accuracy can be controlled with <a href="#">shard_size</a>
Advanced Faceting 	✓ <a href="#">New JSON faceting API as of Solr 5.x</a>	✓ <a href="#">blog post</a>
Geo-distance Faceting	✓	✓
Pivot Facets 	✓	✓
More Like This	✓	✓
Boosting by functions 	✓	✓
Boosting using scripting languages 	✗	✓
Push Queries 	✗ <a href="#">JIRA issue</a>	✓ Percolation. Distributed percolation supported in 1.0
Field collapsing/Results grouping 	✓	✓
Query Re-Ranking 	✓	✓ via <a href="#">Rescoring</a> or <a href="#">a plugin</a>
Index-based Spellcheck 	✓	✓ <a href="#">Phrase Suggester</a>
Wordlist-based Spellcheck 	✓	✗
Autocomplete	✓	✓
Query elevation 	✓	✓ <a href="#">workaround</a>


Feature	Solr 6.2.1	ElasticSearch 5.0
Intra-index joins 	✓ via parent-child query	✓ via <i>has_children</i> and <i>top_children</i> queries
Inter-index joins 	✓ Joined index has to be single-shard and replicated across all nodes.	✗
Resultset Scrolling 	✓ New to 4.7.0	✓ via <i>scan</i> search type
Filter queries 	✓	✓ also supports filtering by native scripts
Filter execution order 	✓ local params and <i>cache</i> property	✓
Alternative QueryParsers 	✓ DisMax, eDisMax	✓ <i>query_string</i> , <i>dis_max</i> , <i>match</i> , <i>multi_match</i> etc
Negative boosting 	✓ but awkward. Involves positively boosting the inverse set of negatively-boosted documents.	✓
Search across multiple indexes	✓ it can search across multiple compatible collections	✓
Result highlighting	✓	✓
Custom Similarity 	✓	✓
Searcher warming on index reload 	✓	✓ <a href="#">Warmers API</a>
Term Vectors API	✓	✓

## Customizability

Feature	Solr 6.2.1	ElasticSearch 5.0
Pluggable API endpoints 	✓	✓
Pluggable search workflow 	✓ via SearchComponents	✗

Feature	Solr 6.2.1	ElasticSearch 5.0
Pluggable update workflow 	✓ via <a href="#">UpdateRequestProcessor</a>	✗
Pluggable Analyzers/Tokenizers	✓	✓
Pluggable QueryParsers 	✓	✓
Pluggable Field Types	✓	✓
Pluggable Function queries	✓	✓
Pluggable scoring scripts	✗	✓
Pluggable hashing 	✓	✓
Pluggable webapps 	✗	✗ [site plugins DEPRECATED in 5.x] <a href="#">blog post</a>
Automated plugin installation 	✗	✓ Installable from GitHub, maven, sonatype or <a href="#">elasticsearch.org</a>

## Distributed

Feature	Solr 6.2.1	ElasticSearch 5.0
Self-contained cluster 	✗ Depends on separate ZooKeeper server	✓ Only Elasticsearch nodes
Automatic node discovery	✓ ZooKeeper	✓ internal Zen Discovery or ZooKeeper
Partition tolerance	✓ The partition without a ZooKeeper quorum will stop accepting indexing requests or cluster state changes, while the partition with a quorum continues to function.	✗ Partitioned clusters can diverge unless <code>discovery.zen.minimum_master_nodes</code> set to at least $N/2+1$ , where N is the size of the cluster. If configured correctly, the partition without a quorum will stop operating, while the other continues to work. See <a href="#">this</a>

Feature	Solr 6.2.1	ElasticSearch 5.0
Automatic failover	✓ If all nodes storing a shard and its replicas fail, client requests will fail, unless requests are made with the <code>shards.tolerant=true</code> parameter, in which case partial results are returned from the available shards.	✓
Automatic leader election	✓	✓
Shard replication	✓	✓
Sharding ?	✓	✓
Automatic shard rebalancing ?	✗	✓ it can be machine, rack, availability zone, and/or data center aware. Arbitrary tags can be assigned to nodes and it can be configured to not assign the same shard and its replicates on a node with the same tags.
Change # of shards	✓ Shards can be added (when using implicit routing) or split (when using <code>compositeld</code> ). Cannot be lowered. Replicas can be increased anytime.	✗ each index has 5 shards by default. Number of primary shards cannot be changed once the index is created. Replicas can be increased anytime.
Shard splitting	✓	✗
Relocate shards and replicas ?	✓ can be done by creating a shard replicate on the desired node and then removing the shard from the source node	✓ can move shards and replicas to any node in the cluster on demand
Control shard routing ?	✓ <code>shards</code> or <code>_route_</code> parameter	✓ <code>routing</code> parameter
Pluggable shard/replica assignment	✓ Rule-based replica assignment	✓ Probabilistic shard balancing with Tempest plugin

Feature	Solr 6.2.1	ElasticSearch 5.0
Consistency	Indexing requests are synchronous with replication. A indexing request won't return until all replicas respond. No check for downed replicas. They will catch up when they recover. When new replicas are added, they won't start accepting and responding to requests until they are finished replicating the index.	Replication between nodes is synchronous by default, thus ES is consistent by default, but it can be set to asynchronous on a per document indexing basis. Index writes can be configured to fail if there are not sufficient active shard replicas. The default is quorum, but all or one are also available.

## Misc

Feature	Solr 6.2.1	ElasticSearch 5.0
Web Admin interface	✔ bundled with Solr	✔ Marvel or Kibana apps
Visualisation	<a href="#">Banana (Port of Kibana)</a>	<a href="#">Kibana</a>
Hosting providers	<a href="#">WebSolr</a> , <a href="#">Searchify</a> , <a href="#">Hosted-Solr</a> , <a href="#">IndexDepot</a> , <a href="#">OpenSolr</a> , <a href="#">gotosolr</a>	<a href="#">Found</a> , <a href="#">Scalefastr</a> , <a href="#">ObjectRocket</a> , <a href="#">bonsai.io</a> , <a href="#">Indexisto</a> , <a href="#">qbox.io</a> , <a href="#">IndexDepot</a> , <a href="#">Compose.io</a> , <a href="#">Sematext</a> <a href="#">Logsene</a>

## Thoughts...

I'm embedding my answer to this "Solr-vs-Elasticsearch" Quora question verbatim here:

1. Elasticsearch was born in the age of REST APIs. If you love REST APIs, you'll probably feel more at home with ES from the get-go. I don't actually think it's 'cleaner' or 'easier to use', but just that it is more aligned with web 2.0 developers' mindsets.
2. Elasticsearch's Query DSL syntax is really flexible and it's pretty easy to write complex queries with it, though it does border on being verbose. Solr doesn't have an equivalent, last I checked. Having said that, I've never found Solr's query syntax wanting, and I've always been able to easily write a custom SearchComponent if needed (more on this later).
3. I find Elasticsearch's documentation to be pretty awful. It doesn't help that some examples



in the documentation are written in YAML and others in JSON. I wrote a ES code parser once to auto-generate documentation from Elasticsearch's source and found a number of discrepancies between code and what's documented on the website, not to mention a number of undocumented/alternative ways to specify the same config key.

By contrast, I've found Solr to be consistent and really well-documented. I've found pretty much everything I've wanted to know about querying and updating indices without having to dig into code much. Solr's `schema.xml` and `solrconfig.xml` are *\*extensively\** documented with most if not all commonly used configurations.

4. Whilst what Rick says about ES being mostly ready to go out-of-box is true, I think that is also a possible problem with ES. Many users don't take the time to do the most simple config (e.g. type mapping) of ES because it 'just works' in dev, and end up running into issues in production.

And once you do have to do config, then I personally prefer Solr's config system over ES'. Long JSON config files can get overwhelming because of the JSON's lack of support for comments. Yes you can use YAML, but it's annoying and confusing to go back and forth between YAML and JSON.

5. If your own app works/thinks in JSON, then without a doubt go for ES because ES thinks in JSON too. Solr merely supports it as an afterthought. ES has a number of nice JSON-related features such as parent-child and nested docs that makes it a very natural fit. Parent-child joins are awkward in Solr, and I don't think there's a Solr equivalent for ES Inner hits.

6. ES doesn't require ZooKeeper for its 'elastic' features which is nice coz I personally find ZK unpleasant, but as a result, ES does have issues with split-brain scenarios though (google 'elasticsearch split-brain' or see this: [Elasticsearch Resiliency Status](#)).

7. Overall from working with clients as a Solr/Elasticsearch consultant, I've found that developer preferences tend to end up along language party lines: if you're a Java/c# developer, you'll be pretty happy with Solr. If you live in Javascript or Ruby, you'll probably love Elasticsearch. If you're on Python or PHP, you'll probably be fine with either.

Something to add about this: ES doesn't have a very elegant Java API IMHO (you'll basically end up using REST because it's less painful), whereas Solrj is very satisfactory and more efficient than Solr's REST API. If you're primarily a Java dev team, do take this into consideration for your sanity. There's no scenario in which constructing JSON in Java is fun/simple, whereas in Python its absolutely pain-free, and believe me, if you have a non-trivial app, your ES json query strings will be works of art.

8. ES doesn't have in-built support for pluggable 'SearchComponents', to use Solr's

terminology. SearchComponents are (for me) a pretty indispensable part of Solr for anyone who needs to do anything customized and in-depth with search queries.

Yes of course, in ES you can just implement your own RestHandler, but that's just not the same as being able to plug-into and rewire the way search queries are handled and parsed.

9. Whichever way you go, I highly suggest you choose a client library which is as 'close to the metal' as you can get. Both ES and Solr have *\*really\** simple search and updating search APIs. If a client library introduces an additional DSL layer in attempt to 'simplify', I suggest you think long and hard about using it, as it's likely to complicate matters in the long-run, and make debugging and asking for help on SO more problematic.

In particular, if you're using Rails + Solr, consider using rsolr/rsolr instead of sunspot/sunspot if you can help it. ActiveRecord is complex code and sufficiently magical. The last thing you want is more magic on top of that.

---

To conclude, ES and Solr have more or less feature-parity and from a feature standpoint, there's rarely one reason to go one way or the other (unless your app lives/breathes JSON). Performance-wise, they are also likely to be quite similar (I'm sure there are exceptions to the rule. ES' relatively new autocomplete implementation, for example, is a pretty dramatic departure from previous Lucene/Solr implementations, and I suspect it produces faster responses at scale).

ES does offer less friction from the get-go and you feel like you have something working much quicker, but I find this to be illusory. Any time gained in this stage is lost when figuring out how to properly configure ES because of poor documentation - an inevitability when you have a non-trivial application.

Solr encourages you to understand a little more about what you're doing, and the chance of you shooting yourself in the foot is somewhat lower, mainly because you're forced to read and modify the 2 well-documented XML config files in order to have a working search app.

---

EDIT on Nov 2015:

ES has been gradually distinguishing itself from Solr when it comes to data analytics. I think it's fair to attribute this to the immense traction of the ELK stack in the logging, monitoring and analytic space. My guess is that this is where Elastic (the company) gets the majority of its revenue, so it makes perfect sense that ES (the product) reflects this.

We see this manifesting primarily in the form of aggregations, which is a more flexible and nuanced replacement for facets. Read more about aggregations here: [Migrating to aggregations](#)

Aggregations have been out for a while now (since 1.4), but with the recently released ES 2.0 comes pipeline aggregations, which let you compute aggregations such as derivatives, moving averages, and series arithmetic on the results of other aggregations. Very cool stuff, and Solr simply doesn't have an equivalent. More on pipeline aggregations here: [Out of this world aggregations](#)

If you're currently using or contemplating using Solr in an analytics app, it is worth your while to look into ES aggregation features to see if you need any of it.

---

## Resources

- My other sites may be of interest if you're new to Lucene, Solr and Elasticsearch:
  - [Lucene Tutorial](#)
  - [Elasticsearch Tutorial](#)
  - [Solr Tutorial](#)
- The [Solr wiki](#) and the [Elasticsearch Guide](#) are your friends.

---

## Contribute

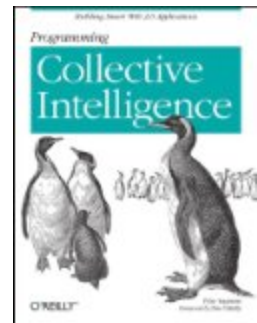
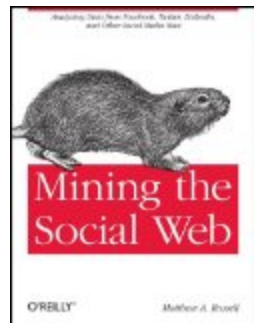
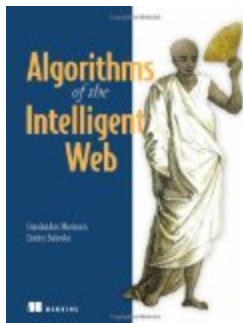
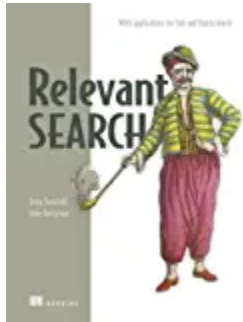
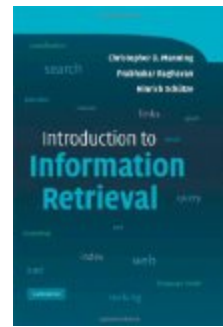
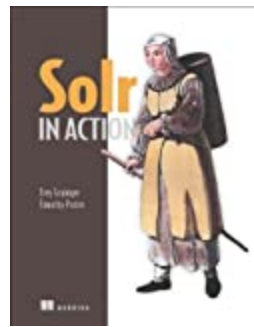
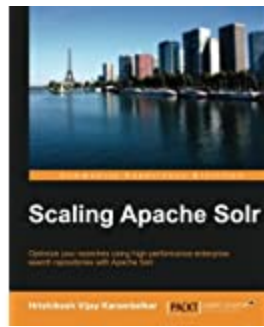
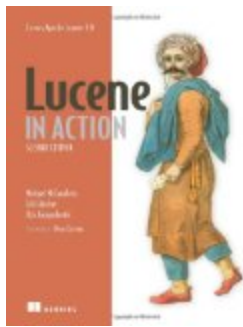
If you see any mistakes, or would like to append to the information on this webpage, you can clone the [GitHub repo for this site](#) with:

```
git clone https://github.com/superkelvint/solr-vs-elasticsearch
```

and submit a pull request.

---

## Popular books related to Search



## Discussion

124 Comments

Solr vs ElasticSearch

[Disqus' Privacy Policy](#)
[1 Login](#)
[Recommend](#) 31

[Tweet](#)
[Share](#)
[Sort by Best](#)


Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS [?](#)

Name



Christian Jensen • 8 years ago

Could you add a comparison of the i18n aspects of each? Specifically how they handle stemming and multiple languages per document?

27 ^ | v • Reply • Share ›



**Peter Corless** → Christian Jensen • 2 years ago

Did you ever get an answer to this? I would believe this would be vital.

4 ^ | v • Reply • Share ›



**Micheal Cooper** → Peter Corless • 2 years ago

This is an important question. Solr out of the box has problems with Japanese, Korean, and Chinese because it requires a CJK parser. For example, older Solr installations included with providers like Pantheon and Acquia is useless without installing external parsers like Kuromoji. How does Elasticsearch perform with CJK languages, which do not have whitespace separation between words?

^ | v • Reply • Share ›



**Ryan Zezeski** • 8 years ago

Just to toot my own horn. I've been working on a project that combines Riak and Solr 4.0. It's called Yokozuna. The goal is to bring the strengths of Riak's distributed bits to Solr and bring Solr's search power to Riak. It does not use SolrCloud at all. Instead Riak manages individual Solr instances and properly shards/replicates/moves data. Below is a link to the Yokozuna 0.1.0 release announcement.

<http://riak-users.197444.n3...>

Below is a link to deploying Yokozuna on EC2 with the pre-built AMI.

<https://github.com/rzezeski...>

Finally, nice comparison. Thanks for making this.

8 ^ | v • Reply • Share ›



**Kelvin Tan** Mod → Ryan Zezeski • 8 years ago

Thanks for posting this Ryan. Added to ES' column under 3rd-party integration.

^ | v • Reply • Share ›



**ehs** → Kelvin Tan • 8 years ago

Wrong column.

1 ^ | v • Reply • Share ›



**Kelvin Tan** Mod → ehs • 8 years ago

Oops. Fixed.

1 ^ | v • Reply • Share ›



**ELevy** → Kelvin Tan • 8 years ago

Not fixed. Still under ES.

^ | v • Reply • Share ›



**Kelvin Tan** Mod → ELevy • 8 years ago

Got reverted in a previous commit. Really fixed now.

^ | v • Reply • Share ›



**Matt Weber** • 8 years ago



[max webb](#) • 8 years ago

Your Elasticsearch section needs some updates. Elasticsearch supports: dynamic fields, field copying via multi-fields, alternative query parsers and pluggable scoring.

12 ^ | ▾ 1 • Reply • Share ›



**Kelvin Tan** Mod ➔ Matt Weber • 8 years ago

@twitter-26266318 : I'm not aware of any alternative query parsers like DisMax. Would you point me to links pls? And I had accidentally inverted the tick/cross in pluggable scoring for solr and ES. Now rectified.

^ | ▾ • Reply • Share ›



**Matt Weber** ➔ Kelvin Tan • 8 years ago

query\_string, dis\_max, match, multi\_match, etc. Plus, you can mix and match all these query types to come up with queries that are much more advanced than what you can do with Solr's edismax.

2 ^ | ▾ • Reply • Share ›



**Kelvin Tan** Mod ➔ Matt Weber • 8 years ago • edited

Thanks! Fixed.

1 ^ | ▾ • Reply • Share ›



**Halvord** • 6 years ago

Elasticsearch does not have a ReST API. It has an API that works over HTTP. Solr's API is kinda-sorts ReSTish but has quirks. One aspect of ReST is that I can send you a search link and it will give you exactly what it gave me. Elasticsearch needs to support search URLs.

About split-brain: read Aphyr on this topic. He made a test harness for doing split-brains and wrote a series of long blog posts about different distributed DBs and their behavior. They're both fantastic and depressing. Nothing did split-brain right except Zookeeper. One could assume that Solr works because it uses Zookeeper. But this is not fair because Zookeeper is not much of a database. Since Solr uses Zookeeper it should work well, if... it uses Zookeeper the right way in all edge cases. Elasticsearch did not support split-brain worth a damn.

<http://aphyr.com/posts/317-...>

<http://aphyr.com/posts/291-...>

4 ^ | ▾ • Reply • Share ›



**Carl-Erik Kopseng** ➔ Halvord • 5 years ago • edited

"One aspect of ReST is that I can send you a search link and it will give you exactly what it gave me. Elasticsearch needs to support search URLs."

Never seen that mentioned in any REST definition. GETs are supposed to be idempotent actions, not changing the resource, but suppose you have a /user/profile url pointing the the users profile. That resource can change over time (editing your name, profile info, etc). So there is no guarantee that what I see is the same you see at some later time.

^ | ▾ • Reply • Share ›



**burtonator** • 3 years ago



We're a recent entrant for Elasticsearch hosting and would love to get a link in your hosting providers section: <https://www.scalefastr.io>

3 ^ | v • Reply • Share ›



**FormerGreenBeret** • 6 years ago

Solr cloud does not support pivot faceting. If you need real time aggregation for OLAP like Queries there is no comparison- Elasticsearch is built to do aggregations. You really should distinguish between solr cloud and regular solr, because IMO you shouldn't compare the functionality of a single node system to a distributed one.

3 ^ | v • Reply • Share ›



**Kelvin Tan** Mod ➔ FormerGreenBeret • 5 years ago

Distributed pivot faceting is now in SolrCloud.

^ | v • Reply • Share ›



**Jayaram Iyer** • 7 years ago

Solr does supports the ability to add New Shards (or DELETE existing shards) to your index (whenever you want) via the "implicit router" configuration (CREATE COLLECTION API).

Lets say - you have to index all "Audit Trail" data of your application into Solr. New Data gets added every day. You might most probably want to shard by year.

You could do something like the below during the initial setup of your collection:

admin/collections?

action=CREATE&

name=AuditTrailIndex&

[router.name=implicit&](#)

shards=2010,2011,2012,2013,2014&

router.field=year

---

[see more](#)

2 ^ | v • Reply • Share ›



**Kelvin Tan** Mod ➔ Jayaram Iyer • 7 years ago

Thanks, changed.

^ | v • Reply • Share ›



**Ryan Zezeski** • 8 years ago • edited

Is it worth mentioning Solr Cell vs. ES's binary attachment support. E.g. ES requires you to base64 encode, Solr requires you to use a separate request handler.

<http://www.elasticsearch.or...>

<http://wiki.apache.org/solr...>

2 ^ | v • Reply • Share ›

**Matt Weber** • 8 years ago

Also, for "change # of shards", this is incorrect. It seems to imply that Solr can change the number of shards on a created index while ElasticSearch can not. This is not true. Solr's numShards parameter is no different than specifying the number of shards while creating an ElasticSearch index and if you started Solr with numShards=5 you would be defaulting to 5 shards just as ElasticSearch does with one difference. That difference is that Solr is not smart enough to "over allocate" shards, so you if start with numShards=5 you better have a 5 node cluster running where as ElasticSearch can allocate those shards on even a single node and move them to new shards as your cluster dynamically grows.

If you really want to get into changing the number of shards, you could always create a new index with the new number of shards and perform a scan query from one index into the new one. Once that is done, update or create an index alias to the old index name and delete the old index.

2 ^ | v 1 • Reply • Share ›

**Jan Høydahl** ➔ Matt Weber • 8 years ago

Shard Splitting is coming to Solr in <https://issues.apache.org/j...> - don't know when though, but then you can increase the number of shards and the system would re-balance automatically by splitting and merging existing shards. Think ES is planning something similar :)

1 ^ | v • Reply • Share ›

**Kelvin Tan** Mod ➔ Matt Weber • 8 years ago

Fixed. Thanks!

^ | v • Reply • Share ›

**Ash Den** • 6 years ago

Solr community has made major strides in strengthening the technology in BIG way!

One of the good read on Apache Solr is - Mastering Apache Solr  
(<http://www.amazon.com/Maste...>

I recently read several SOLR books, brushing up my skills making sure that I was current with the technology. This was the best of the three in this go and the best SOLR resource I have found yet. That said it falls short of the "Mastering" title, in favor of catering more to the beginning user. If I find a better resource, I reserve the option to knock a star off for these reasons.

The books that I read in order of preference were:

Mastering Apache Solr: A practical guide to get to grips with Apache Solr  
Scaling Big Data with Hadoop and Solr (Community Experience Distilled)  
Apache Solr High Performance

While this book needed additional editing and was rife with problems that reflected a lack of editorial oversight, it was by far the best of these three books.

The book provides a complete tutorial with step by step processes to implement everything discussed in the book. There are complete code examples, and even a section providing Java examples of each of the API commands, something lacking even on the Internet



java examples, or each of the API communities, something lacking even on the internet.

1 ^ | v • Reply • Share ›



**Igor Alekseev** • 6 years ago

Complex documents section needs to be updated. As of Solr 4.8 child documents are supported.

<http://heliosearch.org/solr...>

1 ^ | v • Reply • Share ›



**Kelvin Tan** Mod ➔ Igor Alekseev • 5 years ago

Thanks Igor. Updated.

^ | v • Reply • Share ›



**Ismith77** • 8 years ago

Solr has Symfony2 framework integration via <https://github.com/nelmio/N...>, while ElasticSearch has Symfony2 integration via <https://github.com/Exercise...>

1 ^ | v • Reply • Share ›



**Brian Frutchey** • 8 years ago

Does anyone know of good analytics (really Business Intelligence) solutions with deep SOLR or ElasticSearch integration? I am looking for something that will run a query via the search engine, but then perform calculations on the results without needing to replicate data in two different systems or export results (or even long lists of IDs) from one system to another. Like Endeca Information Discovery, only open source!

1 ^ | v • Reply • Share ›



**Kamal M** ➔ Brian Frutchey • 7 years ago

Brian, if you are still looking, take a look at Cloudera's Enterprise Data Hub (EDH). This is exactly the use case - multiple execution engines on top of the SAME data in the same cluster.

^ | v • Reply • Share ›



**Worthy LaFollette** • 8 years ago

Several additions - Foursquare released a DSL called Slashem for querying against SOLR - <https://github.com/foursqua...> ... Also Mongo released a SOLR connector as well. There is also Camel support for SOLR which allows connectivity to AMQ, RabbitMQ, etc..

1 ^ | v • Reply • Share ›



**Ashutosh Bijoor** • 8 years ago

Thanks for putting this together.... really useful... would add tremendous value if you could provide links for each feature to specific sections on the respective documents

1 ^ | v • Reply • Share ›



**Michael Nitschinger** • 8 years ago

Hi,

regarding third-party product integration: Couchbase provides a transport to stream data from a Couchbase Cluster into ElasticSearch: <https://github.com/couchbas...>

Michael

1 ^ | v • Reply • Share ›



**Kelvin Tan** Mod ➔ Michael Nitschinger • 8 years ago

Added. Thanks!

^ | v • Reply • Share ›



**Michael Nitschinger** ➔ Kelvin Tan • 8 years ago

Hey, thanks for adding! I think it would also fit into the "data import" category since that's what the transport is about :)

Best,  
Michael

^ | v • Reply • Share ›



**Jessica Darko** • 8 years ago

Elastic Search can be integrated with Couchbase: [github.com/downloads/couchb...](https://github.com/downloads/couchbase/couchbase-elasticsearch-integration/couchbase-elasticsearch-integration-1.0.0.zip)

1 ^ | v • Reply • Share ›



**Kelvin Tan** Mod ➔ Jessica Darko • 8 years ago

Added. Thanks.

1 ^ | v • Reply • Share ›



**Norbert GC** ➔ Kelvin Tan • 7 years ago

I still cannot see Couchbase in the list, only CouchDB. And they are fundamentally very different products.

^ | v • Reply • Share ›



**Kelvin Tan** Mod ➔ Norbert GC • 7 years ago

Really added this time (to 3rd-party open-source)

^ | v • Reply • Share ›



**Pavel Chumakou** • 8 years ago

Solr supports Pivot facets. Elasticsearch - no.

1 ^ | v • Reply • Share ›



**Kelvin Tan** Mod ➔ Pavel Chumakou • 8 years ago

Added. Thanks!

1 ^ | v • Reply • Share ›



**Y.Kentaro** ➔ Pavel Chumakou • 7 years ago

Pivot Faceting has introduced in elasticsearch 1.0 as aggregation.

c.f.

<http://blog.johtani.info/bl...>

<https://www.found.no/founda...>

^ | v • Reply • Share ›



**FormerGreenBeret** ➔ Y Kentaro • 7 years ago

[Viktor Rotanovs](#) • 8 years ago

solr cloud does not support pivot faceting yet

^ | v • Reply • Share ›

**Viktors Rotanovs** • 8 years ago

Very nice comparison!

How does Elasticsearch compare to Solr in language analysis? E.g. Solr has 3 different analyzers for Polish - is multi-language support generally as strong in Elastic Search?

Also, Solr has several limitations when running in distributed mode, how does Elasticsearch compare in that respect?

1 ^ | v • Reply • Share ›

---

Icons courtesy of [FamFamFam](#)© Copyright 2020 [Kelvin Tan](#) - Solr and Elasticsearch consultant