Elasticsearch vs SOLR Differences and Use cases

Solr has always been more focused on enterprise-directed text searches with advanced information retrieval (IR). Consequently, it’s more suited for search applications that use massive amounts of static data. Solr fits better into enterprise applications that are already implementing big data ecosystem tools, such as Hadoop and Spark.

Elasticsearch is focused more on scaling, data analytics, and processing time series data to obtain meaningful insights and patterns. Elasticsearch is more suited for modern web applications where data is carried in and out in JSON format. Elasticsearch has also put a lot of development effort into making its tool more resilient.

A big difference between Solr and Elasticsearch is that Solr is a completely open-source search engine. Whereas Elasticsearch though open source, is still managed by Elastic’s employees. Solr supports text search, while Elasticsearch is mainly used for analytical querying, filtering, and grouping.

**Searching**

Elasticsearch, DSL is native. The aggregation framework in [Elasticsearch is powerful with aggregation queries](https://logz.io/blog/elasticsearch-queries/) in the APIs have better caching.

### Indexing

Because Elasticsearch is schemaless, it is easy to index unstructured data and dynamic fields without defining the schema of the index in advance.

|  |  |  |
| --- | --- | --- |
| FEATURE | SOLR | ELASTIC SEARCH |
| Installation and Configuration | Easy to get up and running with and very supportive documentation | Easy to get up and running with very supportive documentation. Several packages are available for various platforms. |
| Searching and Indexing | Optimal for text search and enterprise applications close to the big data ecosystem | Useful as both a text search and an analytical engine because of its powerful aggregation module |
| Scalability and Clustering | Support from Solr Cloud and Apache Zookeeper dependence for cluster coordination | Better inherent scalability; design optimal for cloud deployments |
| Community | A historically large ecosystem | A thriving ecosystem for the FOSS version of Elasticsearch and the ELK Stack |
| Documentation | Patchy, out-of-date | Well-documented |

|  |  |  |
| --- | --- | --- |
| FEATURE | SOLR | ELASTIC SEARCH |
| Format | XML, CSV, JSON | JSON |
| HTTP REST API | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Binary API [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png SolrJ | https://solr-vs-elasticsearch.com/img/tick.png TransportClient, Thrift (through a [plugin](https://github.com/elasticsearch/elasticsearch-transport-thrift)) |
| JMX support | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/cross.png ES specific stats are exposed through the REST API |
| Official client libraries [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | Java | Java, Groovy, PHP, Ruby, Perl, Python, .NET, Javascript [Official list of clients](https://www.elastic.co/guide/en/elasticsearch/client/index.html) |
| Community client libraries [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | 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 [Complete list](https://www.elastic.co/guide/en/elasticsearch/client/community/current/index.html) |
| 3rd-party product integration (open-source)[https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | Drupal, Magento, Django, ColdFusion, Wordpress, OpenCMS, Plone, Typo3, ez Publish, Symfony2, Riak (via Yokozuna) | Drupal, Django, Symfony2, Wordpress, CouchBase |
| 3rd-party product integration (commercial)[https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | DataStax Enterprise Search, Cloudera Search, Hortonworks Data Platform, MapR | SearchBlox, Hortonworks Data Platform, MapR etc [Complete list](https://www.elastic.co/guide/en/elasticsearch/plugins/current/integrations.html) |
| Output[https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | JSON, XML, PHP, Python, Ruby, CSV, Velocity, XSLT, native Java | JSON, XML/HTML (via [plugin](http://blog.zenika.com/index.php?post/2012/12/20/Introducing-the-Elasticsearch-View-Plugin)) |

**Infrastructure**

| **Feature** | **SOLR** | **ELASTICSEARCH** |
| --- | --- | --- |
| Master-slave replication | https://solr-vs-elasticsearch.com/img/tick.png **Only in non-SolrCloud.** In SolrCloud, behaves identically to ES. | https://solr-vs-elasticsearch.com/img/cross.png 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** | **ELASTICSEARCH** |
| --- | --- | --- |
| Data Import | DataImportHandler - JDBC, CSV, XML, Tika, URL, Flat File | [DEPRECATED in 2.x] 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 | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| DocValues [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Partial Doc Updates [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png with stored fields | https://solr-vs-elasticsearch.com/img/tick.png with \_source field |
| Custom Analyzers and Tokenizers [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Per-field analyzer chain [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Per-doc/query analyzer chain [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/cross.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Index-time synonyms [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png Supports Solr and Wordnet synonym format |
| Query-time synonyms [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png especially via [hon-lucene-synonyms](https://github.com/healthonnet/hon-lucene-synonyms) | https://solr-vs-elasticsearch.com/img/cross.png Technically, yes, but practically no because multi-word/phrase query-time synonyms are not supported. See [ES docs](https://www.elastic.co/guide/en/elasticsearch/guide/current/multi-word-synonyms.html) and [hon-lucene-synonyms](https://nolanlawson.com/2012/10/31/better-synonym-handling-in-solr/) blog for nuances. |
| Multiple indexes [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Near-Realtime Search/Indexing [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Complex documents [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Schemaless [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png 4.4+ | https://solr-vs-elasticsearch.com/img/tick.png |
| Multiple document types per schema [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/cross.png One set of fields per schema, one schema per core | https://solr-vs-elasticsearch.com/img/tick.png |
| Online schema changes [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png Schemaless mode or via dynamic fields. | https://solr-vs-elasticsearch.com/img/tick.png Only backward-compatible changes. |
| Apache Tika integration [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Dynamic fields [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Field copying [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png via multi-fields |
| Hash-based deduplication [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png [Murmur plugin](https://www.elastic.co/guide/en/elasticsearch/plugins/current/mapper-murmur3.html) or [ER plugin](https://github.com/YannBrrd/elasticsearch-entity-resolution) |

**Searching**

| **Feature** | **SOLR** | **ELASTICSEARCH** |
| --- | --- | --- |
| Lucene Query parsing [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Structured Query DSL [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/cross.png Need to programmatically create queries if going beyond Lucene query syntax. | https://solr-vs-elasticsearch.com/img/tick.png |
| Span queries [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png via [SOLR-2703](https://issues.apache.org/jira/browse/SOLR-2703) | https://solr-vs-elasticsearch.com/img/tick.png |
| Spatial/geo search [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Multi-point spatial search [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Faceting [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png Top N term accuracy can be controlled with [shard\_size](https://www.elastic.co/guide/en/elasticsearch/reference/2.1/search-aggregations-bucket-terms-aggregation.html" \l "_shard_size_2) |
| Advanced Faceting [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png [New JSON faceting API as of Solr 5.x](http://yonik.com/json-facet-api/) | https://solr-vs-elasticsearch.com/img/tick.png [blog post](http://www.elasticsearch.org/blog/data-visualization-elasticsearch-aggregations) |
| Geo-distance Faceting | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Pivot Facets [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| More Like This | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Boosting by functions [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Boosting using scripting languages [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/cross.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Push Queries [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/cross.png[JIRA issue](https://issues.apache.org/jira/browse/SOLR-4587) | https://solr-vs-elasticsearch.com/img/tick.png Percolation. Distributed percolation supported in 1.0 |
| Field collapsing/Results grouping [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Query Re-Ranking [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png via [Rescoring](https://www.elastic.co/guide/en/elasticsearch/reference/current/search-request-rescore.html) or [a plugin](https://github.com/codelibs/elasticsearch-dynarank) |
| Index-based Spellcheck [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png [Phrase Suggester](https://www.elastic.co/guide/en/elasticsearch/reference/current/search-suggesters-phrase.html) |
| Wordlist-based Spellcheck [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/cross.png |
| Autocomplete | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Query elevation [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png[workaround](https://github.com/elasticsearch/elasticsearch/issues/1066#issuecomment-8625739) |
| Intra-index joins [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png via parent-child query | https://solr-vs-elasticsearch.com/img/tick.png via *has\_children* and *top\_children* queries |
| Inter-index joins [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png Joined index has to be single-shard and replicated across all nodes. | https://solr-vs-elasticsearch.com/img/cross.png |
| Resultset Scrolling [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png New to 4.7.0 | https://solr-vs-elasticsearch.com/img/tick.png via *scan* search type |
| Filter queries [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png also supports filtering by native scripts |
| Filter execution order [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png local params and *cache* property | https://solr-vs-elasticsearch.com/img/tick.png |
| Alternative QueryParsers [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png DisMax, eDisMax | https://solr-vs-elasticsearch.com/img/tick.png query\_string, dis\_max, match, multi\_match etc |
| Negative boosting [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png But awkward. Involves positively boosting the inverse set of negatively-boosted documents. | https://solr-vs-elasticsearch.com/img/tick.png |
| Search across multiple indexes | https://solr-vs-elasticsearch.com/img/tick.png it can search across multiple compatible collections | https://solr-vs-elasticsearch.com/img/tick.png |
| Result highlighting | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Custom Similarity [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Searcher warming on index reload [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png [Warmers API](http://www.elasticsearch.org/guide/reference/api/admin-indices-warmers/) |
| Term Vectors API | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |

**Customizability**

| **Feature** | **SOLR** | **ELASTICSEARCH** |
| --- | --- | --- |
| Pluggable API endpoints [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Pluggable search workflow [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png via SearchComponents | https://solr-vs-elasticsearch.com/img/cross.png |
| Pluggable update workflow [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png via [UpdateRequestProcessor](https://cwiki.apache.org/confluence/display/solr/Update+Request+Processors" \t "_blank) | https://solr-vs-elasticsearch.com/img/cross.png |
| Pluggable Analyzers/Tokenizers | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Pluggable QueryParsers [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Pluggable Field Types | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Pluggable Function queries | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Pluggable scoring scripts | https://solr-vs-elasticsearch.com/img/cross.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Pluggable hashing [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Pluggable webapps [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/cross.png | https://solr-vs-elasticsearch.com/img/cross.png [site plugins DEPRECATED in 5.x] [blog post](https://www.elastic.co/blog/running-site-plugins-with-elasticsearch-5-0) |
| Automated plugin installation [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/cross.png | https://solr-vs-elasticsearch.com/img/tick.png Installable from GitHub, maven, sonatype or elasticsearch.org |

**Distributed**

| **Feature** | **SOLR** | **ELASTICSEARCH** |
| --- | --- | --- |
| Self-contained cluster [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/cross.png Depends on separate ZooKeeper server | https://solr-vs-elasticsearch.com/img/tick.png Only Elasticsearch nodes |
| Automatic node discovery | https://solr-vs-elasticsearch.com/img/tick.png ZooKeeper | https://solr-vs-elasticsearch.com/img/tick.png internal Zen Discovery or ZooKeeper |
| Partition tolerance | https://solr-vs-elasticsearch.com/img/tick.png The partition without a ZooKeeper quorum will stop accepting indexing requests or cluster state changes, while the partition with a quorum continues to function. | https://solr-vs-elasticsearch.com/img/cross.png Partitioned clusters can diverge unless discovery.zen.minimum\_master\_nodes 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 [this](http://elasticsearch-users.115913.n3.nabble.com/Split-brain-td3620149.html) |
| Automatic failover | https://solr-vs-elasticsearch.com/img/tick.png If all nodes storing a shard and its replicas fail, client requests will fail, unless requests are made with the shards.tolerant=true parameter, in which case partial results are retuned from the available shards. | https://solr-vs-elasticsearch.com/img/tick.png |
| Automatic leader election | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Shard replication | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Sharding [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/tick.png |
| Automatic shard rebalancing [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/cross.png | https://solr-vs-elasticsearch.com/img/tick.png 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 | https://solr-vs-elasticsearch.com/img/tick.png Shards can be added (when using implicit routing) or split (when using compositeId). Cannot be lowered. Replicas can be increased anytime. | https://solr-vs-elasticsearch.com/img/cross.png 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 | https://solr-vs-elasticsearch.com/img/tick.png | https://solr-vs-elasticsearch.com/img/cross.png |
| Relocate shards and replicas [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png can be done by creating a shard replicate on the desired node and then removing the shard from the source node | https://solr-vs-elasticsearch.com/img/tick.png can move shards and replicas to any node in the cluster on demand |
| Control shard routing [https://solr-vs-elasticsearch.com/img/help.png](https://solr-vs-elasticsearch.com/) | https://solr-vs-elasticsearch.com/img/tick.png *shards* or *\_route\_* parameter | https://solr-vs-elasticsearch.com/img/tick.png *routing* parameter |
| Pluggable shard/replica assignment | https://solr-vs-elasticsearch.com/img/tick.png [Rule-based replica assignment](https://cwiki.apache.org/confluence/display/solr/Rule-based+Replica+Placement) | https://solr-vs-elasticsearch.com/img/tick.png Probabilistic shard balancing with [Tempest plugin](https://github.com/datarank/tempest) |
| 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 is there are not sufficient active shard replicas. The default is quorum, but all or one are also available. |

**Misc**

| **Feature** | **SOLR** | **ELASTICSEARCH** |
| --- | --- | --- |
| Web Admin interface | https://solr-vs-elasticsearch.com/img/tick.png bundled with Solr | https://solr-vs-elasticsearch.com/img/tick.png Marvel or Kibana apps |
| Visualization | [Banana (Port of Kibana)](https://github.com/LucidWorks/banana) | [Kibana](https://www.elastic.co/products/kibana) |
| Hosting providers | [WebSolr](http://www.websolr.com/), [Searchify](http://www.searchify.com/), [Hosted-Solr](http://www.hosted-solr.com/), [IndexDepot](http://www.indexdepot.com/), [OpenSolr](http://www.opensolr.com/), [gotosolr](http://www.gotosolr.com/) | [Found](https://www.elastic.co/found), [Scalefastr](https://www.scalefastr.io/), [ObjectRocket](http://objectrocket.com/elasticsearch/), [bonsai.io](http://www.bonsai.io/), [Indexisto](http://www.indexisto.com/), [qbox.io](http://www.qbox.io/), [IndexDepot](http://www.indexdepot.com/), [Compose.io](http://www.compose.io/), [Sematext Logsene](https://www.sematext.com/logsene) |

**Deep dive into Elastic Search**

**What is Elasticsearch?**

Elasticsearch is a distributed, open-source search and analytics engine built on Apache Lucene and developed in Java. It started as a scalable version of the Lucene open-source search framework then added the ability to horizontally scale Lucene indices. Elasticsearch allows you to store, search, and analyze huge volumes of data quickly and in near real-time and give back answers in milliseconds. It’s able to achieve fast search responses because instead of searching the text directly, it searches an index. It uses a structure based on documents instead of tables and schemas and comes with extensive REST APIs for storing and searching the data. At its core, you can think of Elasticsearch as a server that can process JSON requests and give you back JSON data.

**How does Elasticsearch work?**

**Logical Concepts**

**Documents**

Documents are the basic unit of information that can be indexed in Elasticsearch expressed in JSON, which is the global internet data interchange format. You can think of a document like a row in a relational database, representing a given entity — the thing you’re searching for. In Elasticsearch, a document can be more than just text, it can be any structured data encoded in JSON. That data can be things like numbers, strings, and dates. Each document has a unique ID and a given data type, which describes what kind of entity the document is. For example, a document can represent an encyclopedia article or log entries from a web server.

**Indices**

An index is a collection of documents that have similar characteristics. An index is the highest level entity that you can query against in Elasticsearch. You can think of the index as being similar to a database in a relational database schema. Any documents in an index are typically logically related. In the context of an e-commerce website, for example, you can have an index for Customers, one for Products, one for Orders, and so on. An index is identified by a name that is used to refer to the index while performing indexing, search, update, and delete operations against the documents in it.

**Inverted Index**

An index in Elasticsearch is actually what’s called an inverted index, which is the mechanism by which all search engines work. It is a data structure that stores a mapping from content, such as words or numbers, to its locations in a document or a set of documents. Basically, it is a hashmap-like data structure that directs you from a word to a document. An inverted index doesn’t store strings directly and instead splits each document up to individual search terms (i.e. each word) then maps each search term to the documents those search terms occur within. For example, in the image below, the term “best” occurs in document 2, so it is mapped to that document. This serves as a quick look-up of where to find search terms in a given document. By using distributed inverted indices, Elasticsearch quickly finds the best matches for full-text searches from even very large data sets.

**Cluster**

An Elasticsearch cluster is a group of one or more node instances that are connected together. The power of an Elasticsearch cluster lies in the distribution of tasks, searching, and indexing, across all the nodes in the cluster.

**Node**

A node is a single server that is a part of a cluster. A node stores data and participates in the cluster’s indexing and search capabilities. An Elasticsearch node can be configured in different ways:

**Master Node:** Controls the Elasticsearch cluster and is responsible for all cluster-wide operations like creating/deleting an index and adding/removing nodes.

**Data Node:** Stores data and executes data-related operations such as search and aggregation.

**Client Node:** Forwards cluster requests to the master node and data-related requests to data nodes.

**Shards**

Elasticsearch provides the ability to subdivide the index into multiple pieces called shards. Each shard is in itself a fully-functional and independent “index” that can be hosted on any node within a cluster. By distributing the documents in an index across multiple shards, and distributing those shards across multiple nodes, Elasticsearch can ensure redundancy, which both protects against hardware failures and increases query capacity as nodes are added to a cluster.

**Replicas**

Elasticsearch allows you to make one or more copies of your index’s shards which are called “replica shards” or just “replicas”. Basically, a replica shard is a copy of a primary shard. Each document in an index belongs to one primary shard. Replicas provide redundant copies of your data to protect against hardware failure and increase capacity to serve read requests like searching or retrieving a document.

**The Elastic Stack (ELK)**

Elasticsearch is the central component of the Elastic Stack, a set of open-source tools for data ingestion, enrichment, storage, analysis, and visualization. It is commonly referred to as the “ELK” stack after its components Elasticsearch, Logstash, and Kibana and now also includes Beats. Although a search engine at its core, users started using Elasticsearch for log data and wanted a way to easily ingest and visualize that data.

**Kibana**

Kibana is a data visualization and management tool for Elasticsearch that provides real-time histograms, line graphs, pie charts, and maps. It lets you visualize your Elasticsearch data and navigate the Elastic Stack. You can select the way you give shape to your data by starting with one question to find out where the interactive visualization will lead you. For example, since Kibana is often used for log analysis, it allows you to answer questions about where your web hits are coming from, your distribution URLs, and so on. If you’re not building your own application on top of Elasticsearch, Kibana is a great way to search and visualize your index with a powerful and flexible UI. However, a major drawback is that every visualization can only work against a single index/index pattern. So if you have indices with strictly different data, you’ll have to create separate visualizations for each. For more advanced use cases, Knowi is a good option. It allows you to join your Elasticsearch data across multiple indexes and blend it with other SQL/NoSQL/REST-API data sources, then create visualizations from it in a business-user friendly UI.

**Logstash**

Logstash is used to aggregate and process data and send it to Elasticsearch. It is an open-source, server-side data processing pipeline that ingests data from a multitude of sources simultaneously, transforms it, and then sends it to collect. It also transforms and prepares data regardless of format by identifying named fields to build structure, and transform them to converge on a common format. For example, since data is often scattered across different systems in various formats, Logstash allows you to tie different systems together like web servers, databases, Amazon services, etc. and publish data to wherever it needs to go in a continuous streaming fashion.

**Beats**

Beats is a collection of lightweight, single-purpose data shipping agents used to send data from hundreds or thousands of machines and systems to Logstash or Elasticsearch. Beats are great for gathering data as they can sit on your servers, with your containers, or deploy as functions then centralize data in Elasticsearch. For example, Filebeat can sit on your server, monitor log files as they come in, parses them, and import into Elasticsearch in near-real-time.

**What is Elasticsearch used for?**

Now that we have a general understanding of what Elasticsearch is, the logical concepts behind it, and its architecture, we have a better sense of why and how it can be used for a variety of use cases. Below, we’ll examine some of Elasticsearch’s primary use cases and provide examples of how companies are using it today.

**Primary Use Cases**

**Application search:** For applications that rely heavily on a search platform for the access, retrieval, and reporting of data.

**Website search:** Websites which store a lot of content find Elasticsearch a very useful tool for effective and accurate searches. It’s no surprise that Elasticsearch is steadily gaining ground in the site search domain sphere.

**Enterprise search:** Elasticsearch allows enterprise-wide search that includes document search, E-commerce product search, blog search, people search, and any form of search you can think of. In fact, it has steadily penetrated and replaced the search solutions of most of the popular websites we use on a daily basis. From a more enterprise-specific perspective, Elasticsearch is used to great success in company intranets.

**Logging and log analytics:** As we’ve discussed, Elasticsearch is commonly used for ingesting and analyzing log data in near-real-time and in a scalable manner. It also provides important operational insights on log metrics to drive actions.

**Infrastructure metrics and container monitoring:** Many companies use the ELK stack to analyze various metrics. This may involve gathering data across several performance parameters that vary by use case.

**Security analytics**: Another major analytics application of Elasticsearch is security analysis. Access logs and similar logs concerning system security can be analyzed with the ELK stack, providing a more complete picture of what’s going on across your systems in real-time.

**Business analytics:** Many of the built-in features available within the ELK Stack makes it a good option as a business analytics tool. However, there is a steep learning curve for implementing this product and in most organizations. This is especially true in cases where companies have multiple data sources besides Elasticsearch–since Kibana only works with Elasticsearch data. A good alternative is Knowi, an analytics platform that natively integrates with Elasticsearch and allows even non-technical business users to create visualizations and perform analytics on Elasticsearch data without prior knowledge or expertise of the ELK Stack.

**Elastic Search on AWS**

For AWS elastic search amazon takes elastic search branch and develops on their own, as Elastic search updates are handled by Elastic team employees.

Amazon Elastic search Service lets you store up to 3 PB of data in a single cluster, enabling you to run large log analytics workloads via a single Kibana interface.

Easily scalable, highly secure and cost effective. (Amazon Elasticsearch Service lets you pay only for what you use – there are no upfront costs or usage requirements)

**References:**

Solr vs. Elasticsearch: <https://logz.io/blog/solr-vs-elasticsearch/>

GitHub Elasticsearch: [https://github.com/elastic/elasticsearch](https://github.com/elastic/elasticsearch-transport-thrift)

Elasticsearch: <https://www.elastic.co/guide/index.html>

Blog Zenika: [https://blog.zenika.com](https://blog.zenika.com/?post%2F2012%2F12%2F20%2FIntroducing-the-Elasticsearch-View-Plugin)

Read the Tea Leaves: [https://nolanlawson.com](https://nolanlawson.com/2012/10/31/better-synonym-handling-in-solr/)

Apache Open Issues: [https://issues.apache.org/jira/browse/](https://issues.apache.org/jira/browse/SOLR-2703)

Advanced Faceting: <http://yonik.com/json-facet-api/>

Elastic: <https://www.elastic.co/>

Solr Ref Guide 6.6: <https://lucene.apache.org/solr/guide/6_6/update-request-processors.html>

Deep dive into Elastic Search: <https://www.knowi.com/blog/what-is-elastic-search/>