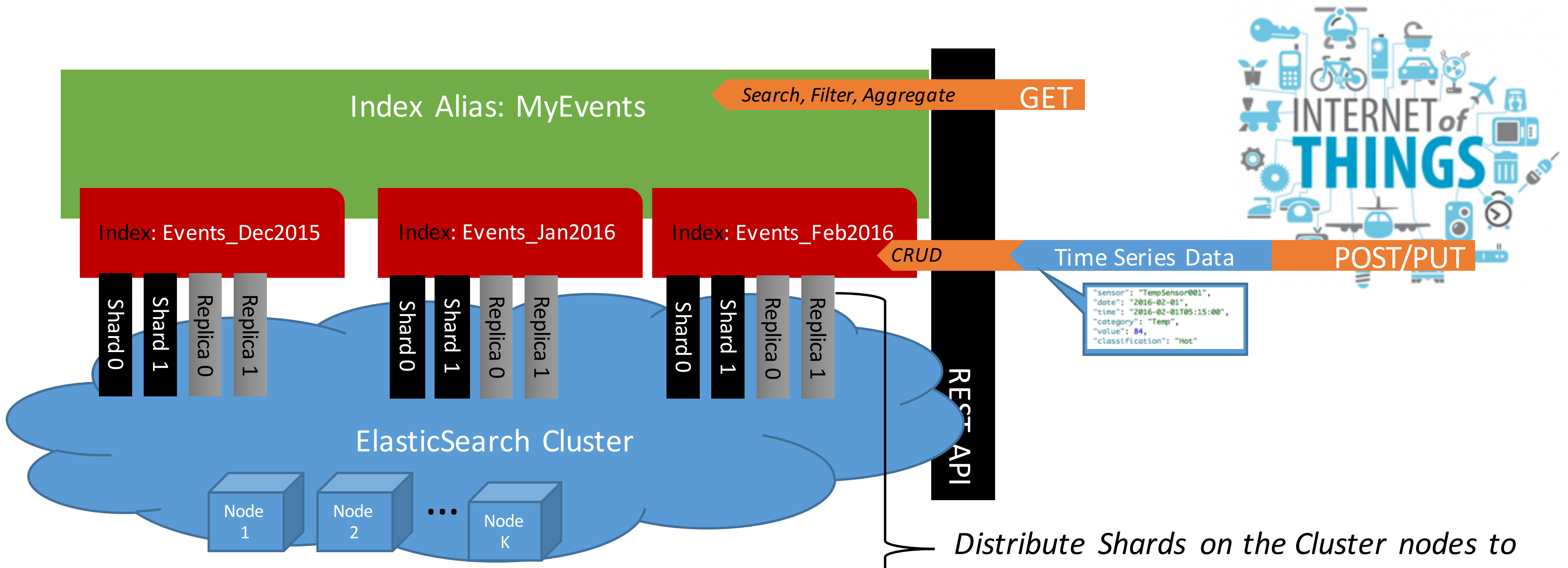
ElasticSearch Concepts



```
"sensor": "TempSensor001",
"date": "2016-02-01",
"time": "2016-02-01T05:15:00",
"category": "Temp",
"value": 84,
"classification": "Hot"
```

Distribute Shards on the Cluster nodes to achieve scalability as Indices grow

- Subdivide your index into multiple pieces called shards
- When you create an index, you can simply define the number of shards that you want
- Each shard is in itself a fully-functional and independent "index" that can be hosted on any node in the cluster
- This allows you to horizontally split/scale your content volume (Scalability)
- This allows you to distribute and parallelize operations across shards (on multiple nodes)
- Elasticsearch allows you to make one or more copies of your index’s shards into replica shards (high availability)
- Searches can be executed on all replicas in parallel
- After the index is created, you may change the number of replicas dynamically anytime but you cannot change the number shards
- Indices can be logically grouped by an Alias

Index and Alias Creation

Base Address: http://192.168.1.95:9200

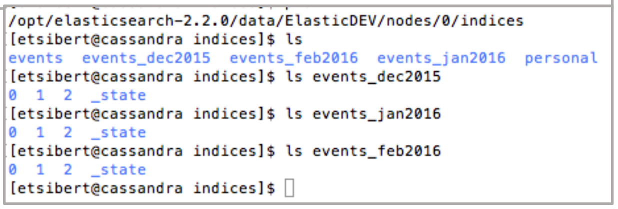
```
PUT /events_dec2015
{
"settings":{
    "index":{
            "number_of_shards":3,
            "number_of_replicas":2
          }
        }
}
```

```
PUT /events_jan2016
{
"settings":{
    "index":{
            "number_of_shards":3,
            "number_of_replicas":2
          }
        }
}
```

```
PUT /events_feb2016
{
"settings":{
    "index":{
            "number_of_shards":3,
            "number_of_replicas":2
          }
        }
}
```

```
POST /_aliases
{
   "actions":[
            {"add":{"index":"events_dec2015", "alias":"myevents"}},
            {"add":{"index":"events_jan2016", "alias":"myevents"}},
            {"add":{"index":"events_feb2016", "alias":"myevents"}}
           ]
}
```

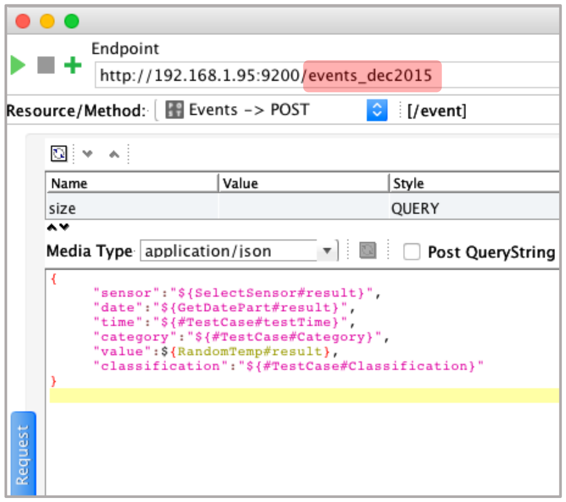
Physical Shard Storage



```
/opt/elasticsearch-2.2.0/data/ElasticDEV/nodes/0/indices
[etsibert@cassandra indices]$ ls
events  events_dec2015  events_feb2016  events_jan2016  personal
[etsibert@cassandra indices]$ ls events_dec2015
0  1  2  _state
[etsibert@cassandra indices]$ ls events_jan2016
0  1  2  _state
[etsibert@cassandra indices]$ ls events_feb2016
0  1  2  _state
[etsibert@cassandra indices]$ 
```

Data WRITE (data generation with SOAPUI)

December 2015



Endpoint
http://192.168.1.95:9200/events_dec2015

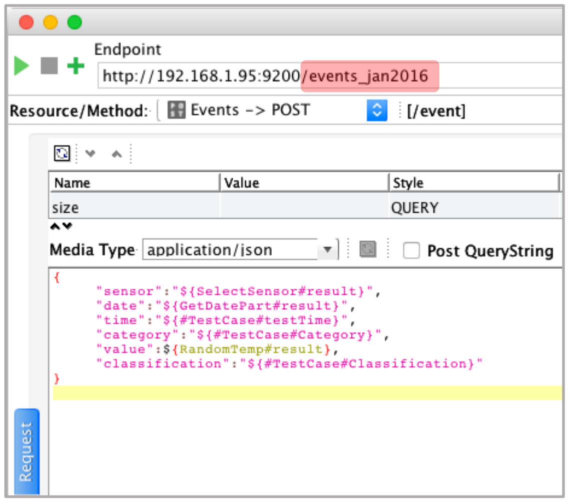
Resource/Method: Events -> POST [/event]

Name	Value	Style
size		QUERY

Media Type: application/json ☐ Post QueryString

```
{
      "sensor":"${SelectSensor#result}",
      "date":"${GetDatePart#result}",
      "time":"${#TestCase#testTime}",
      "category":"${#TestCase#Category}",
      "value":${RandomTemp#result},
      "classification":"${#TestCase#Classification}"
}
```

January 2016



Endpoint
http://192.168.1.95:9200/events_jan2016

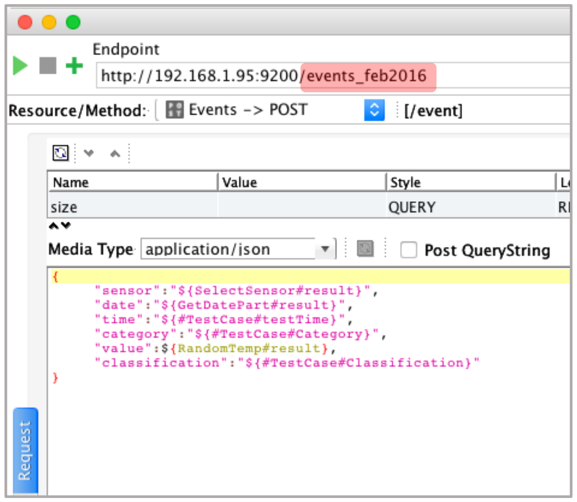
Resource/Method: Events -> POST [/event]

Name	Value	Style
size		QUERY

Media Type: application/json ☐ Post QueryString

```
{
      "sensor":"${SelectSensor#result}",
      "date":"${GetDatePart#result}",
      "time":"${#TestCase#testTime}",
      "category":"${#TestCase#Category}",
      "value":${RandomTemp#result},
      "classification":"${#TestCase#Classification}"
}
```

February 2016



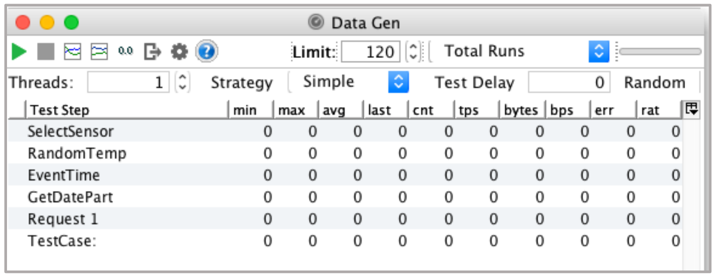
Endpoint
http://192.168.1.95:9200/events_feb2016

Resource/Method: Events -> POST [/event]

Name	Value	Style
size		QUERY

Media Type: application/json ☐ Post QueryString

```
{
      "sensor":"${SelectSensor#result}",
      "date":"${GetDatePart#result}",
      "time":"${#TestCase#testTime}",
      "category":"${#TestCase#Category}",
      "value":${RandomTemp#result},
      "classification":"${#TestCase#Classification}"
}
```



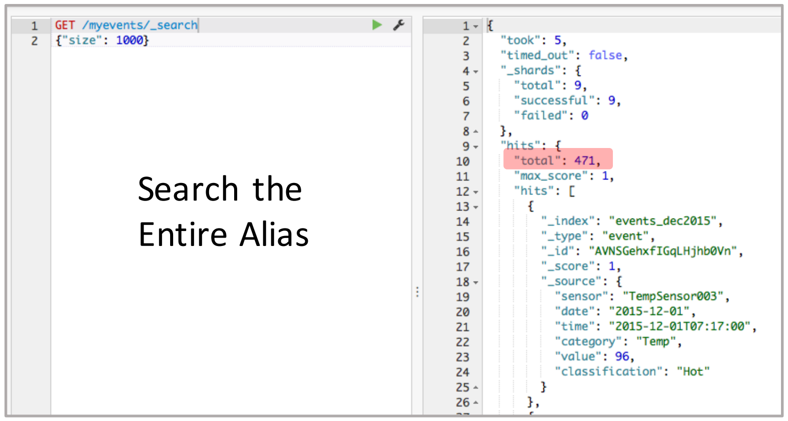
Data Gen

Limit: 120 Total Runs

Threads: 1 Strategy Simple Test Delay 0 Random

Test Step	min	max	avg	last	cnt	tps	bytes	bps	err	rat
SelectSensor	0	0	0	0	0	0	0	0	0	0
RandomTemp	0	0	0	0	0	0	0	0	0	0
EventTime	0	0	0	0	0	0	0	0	0	0
GetDatePart	0	0	0	0	0	0	0	0	0	0
Request 1	0	0	0	0	0	0	0	0	0	0
TestCase:	0	0	0	0	0	0	0	0	0	0

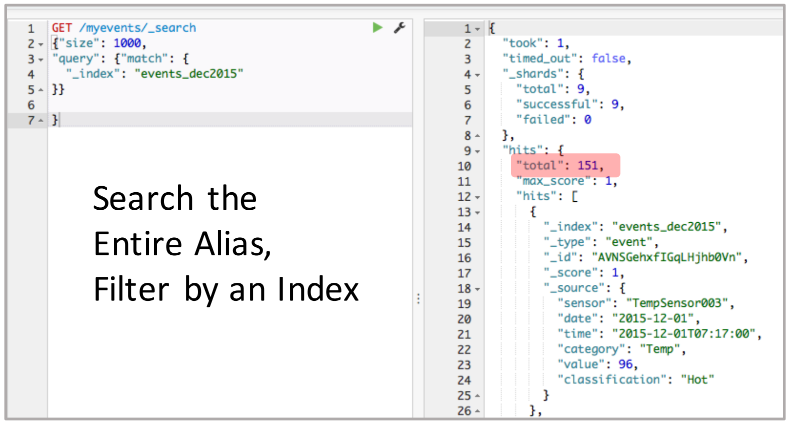
Search API



Search the
Entire Alias

```
GET /myevents/_search
{"size": 1000}
```

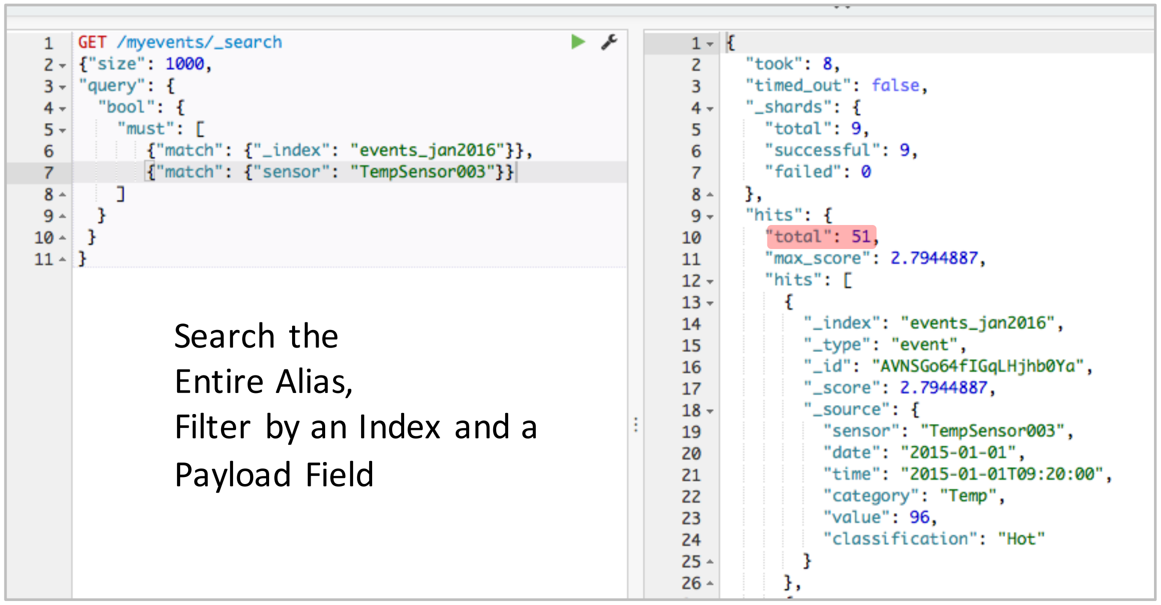
```
{
  "took": 5,
  "timed_out": false,
  "_shards": {
    "total": 9,
    "successful": 9,
    "failed": 0
  },
  "hits": {
    "total": 471,
    "max_score": 1,
    "hits": [
      {
        "_index": "events_dec2015",
        "_type": "event",
        "_id": "AVNSGehxfIGqLHjhb0Vn",
        "_score": 1,
        "_source": {
          "sensor": "TempSensor003",
          "date": "2015-12-01",
          "time": "2015-12-01T07:17:00",
          "category": "Temp",
          "value": 96,
          "classification": "Hot"
        }
      },
```



Search the
Entire Alias,
Filter by an Index

```
GET /myevents/_search
{"size": 1000,
"query": {"match": {
  "_index": "events_dec2015"
}}

}
```

```
{
  "took": 1,
  "timed_out": false,
  "_shards": {
    "total": 9,
    "successful": 9,
    "failed": 0
  },
  "hits": {
    "total": 151,
    "max_score": 1,
    "hits": [
      {
        "_index": "events_dec2015",
        "_type": "event",
        "_id": "AVNSGehxfIGqLHjhb0Vn",
        "_score": 1,
        "_source": {
          "sensor": "TempSensor003",
          "date": "2015-12-01",
          "time": "2015-12-01T07:17:00",
          "category": "Temp",
          "value": 96,
          "classification": "Hot"
        }
      },
```

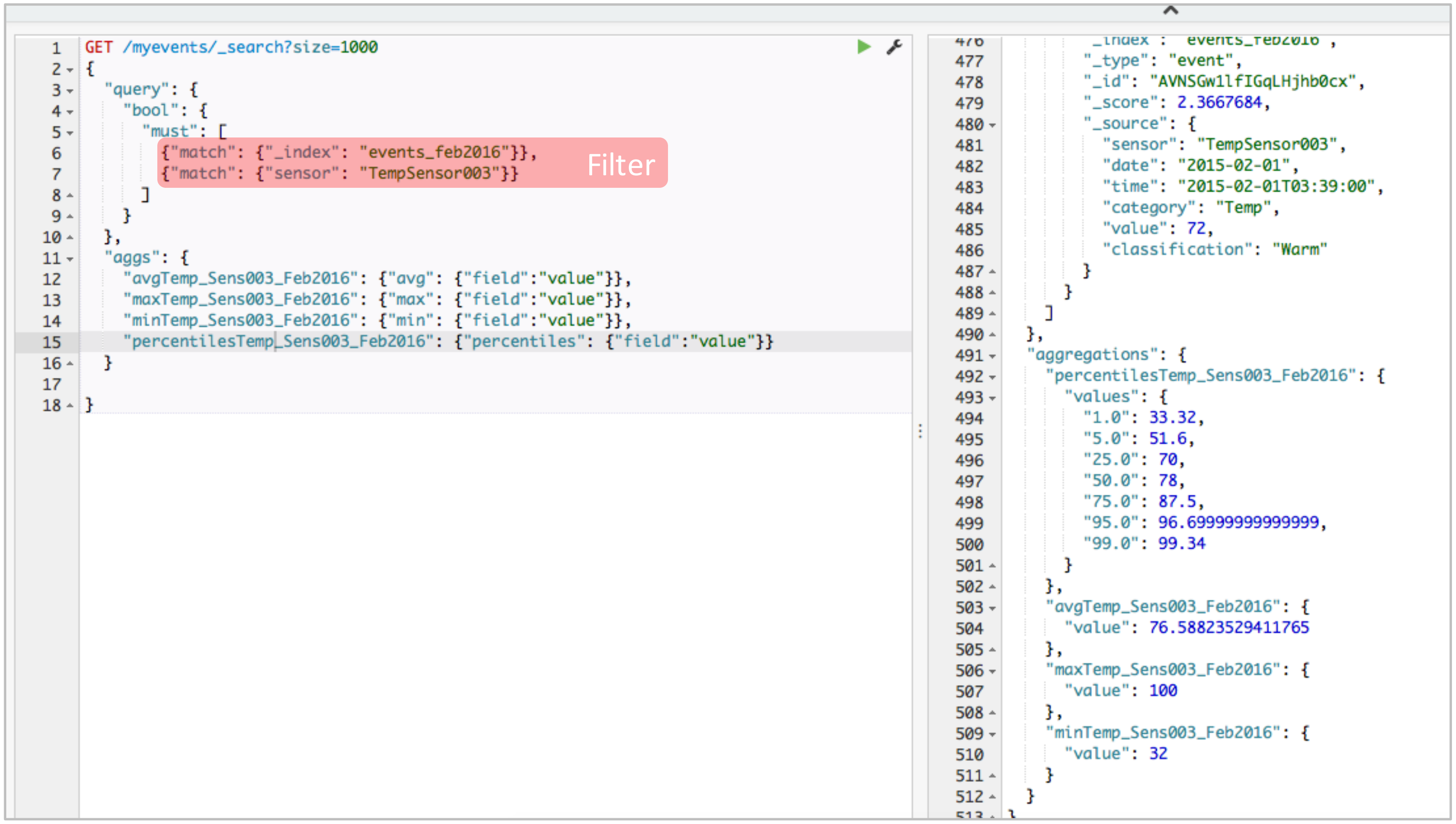


Search the
Entire Alias,
Filter by an Index and a
Payload Field

```
GET /myevents/_search
{"size": 1000,
"query": {
  "bool": {
    "must": [
      {"match": {"_index": "events_jan2016"}},
      {"match": {"sensor": "TempSensor003"}}
    ]
  }
 }
}
```

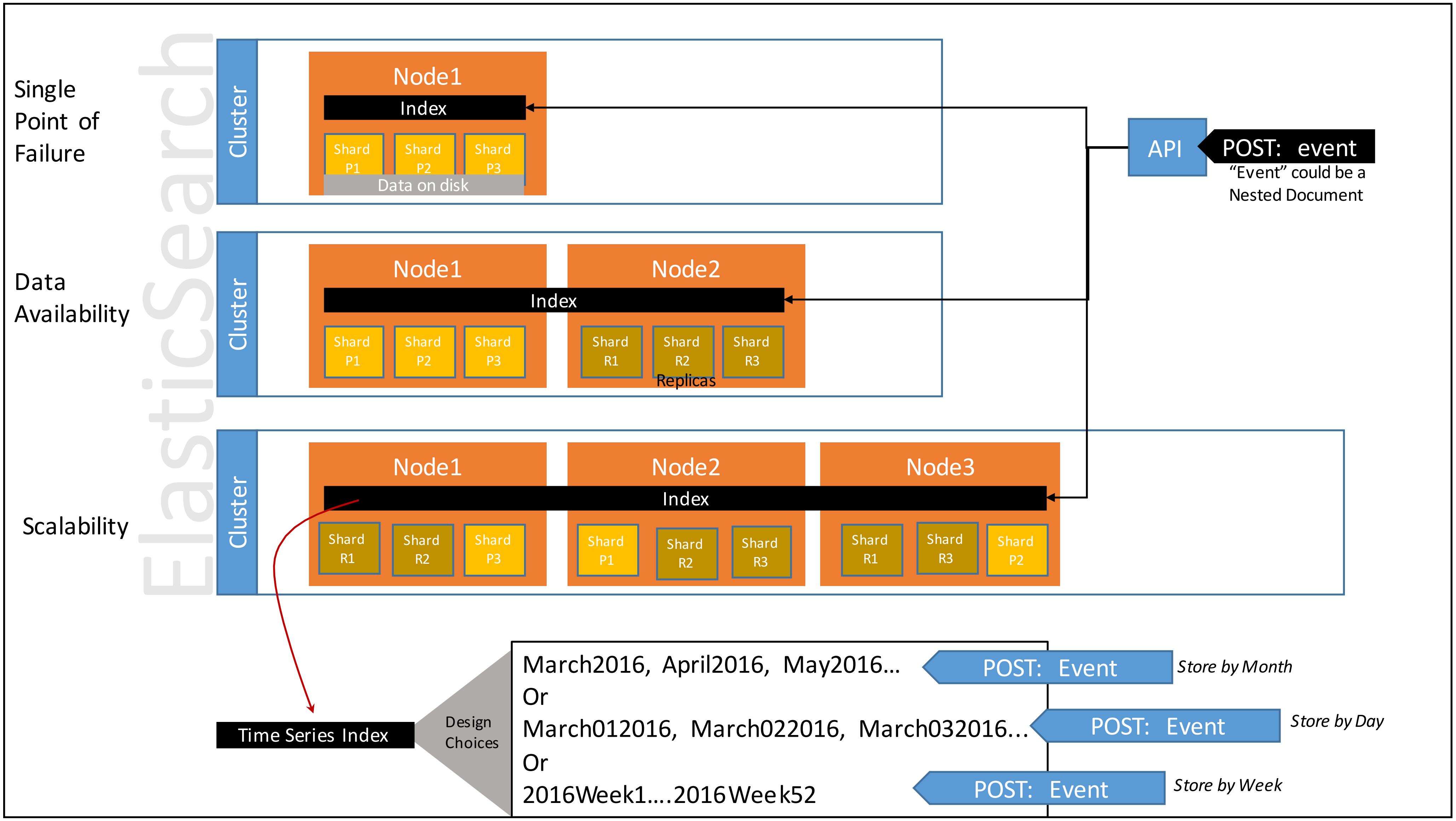
```
{
  "took": 8,
  "timed_out": false,
  "_shards": {
    "total": 9,
    "successful": 9,
    "failed": 0
  },
  "hits": {
    "total": 51,
    "max_score": 2.7944887,
    "hits": [
      {
        "_index": "events_jan2016",
        "_type": "event",
        "_id": "AVNSGo64fIGqLHjhb0Ya",
        "_score": 2.7944887,
        "_source": {
          "sensor": "TempSensor003",
          "date": "2015-01-01",
          "time": "2015-01-01T09:20:00",
          "category": "Temp",
          "value": 96,
          "classification": "Hot"
        }
      },
```

Aggregation API



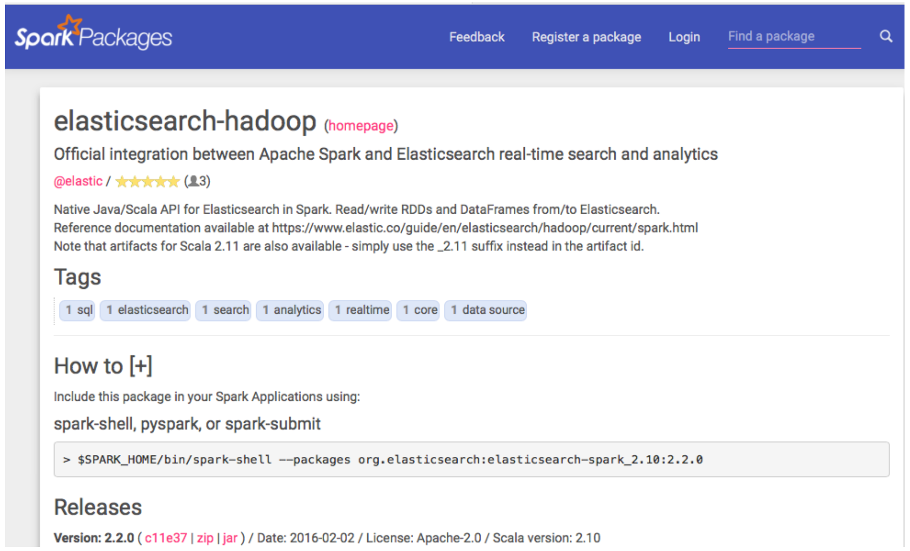
```
GET /myevents/_search?size=1000
{
  "query": {
    "bool": {
      "must": [
        {"match": {"_index": "events_feb2016"}},
        {"match": {"sensor": "TempSensor003"}}
      ]
    }
  },
  "aggs": {
    "avgTemp_Sens003_Feb2016": {"avg": {"field":"value"}},
    "maxTemp_Sens003_Feb2016": {"max": {"field":"value"}},
    "minTemp_Sens003_Feb2016": {"min": {"field":"value"}},
    "percentilesTemp_Sens003_Feb2016": {"percentiles": {"field":"value"}}
  }

}
```

```
        "_type": "event",
        "_id": "AVNSGw1lfIGqLHjhb0cx",
        "_score": 2.3667684,
        "_source": {
          "sensor": "TempSensor003",
          "date": "2015-02-01",
          "time": "2015-02-01T03:39:00",
          "category": "Temp",
          "value": 72,
          "classification": "Warm"
        }
      }
    ]
  },
  "aggregations": {
    "percentilesTemp_Sens003_Feb2016": {
      "values": {
        "1.0": 33.32,
        "5.0": 51.6,
        "25.0": 70,
        "50.0": 78,
        "75.0": 87.5,
        "95.0": 96.69999999999999,
        "99.0": 99.34
      }
    },
    "avgTemp_Sens003_Feb2016": {
      "value": 76.58823529411765
    },
    "maxTemp_Sens003_Feb2016": {
      "value": 100
    },
    "minTemp_Sens003_Feb2016": {
      "value": 32
    }
  }
}
```


Single Point of Failure
Data Availability
Scalability
ElasticSearch
Cluster
Node1
Index
Shard P1
Shard P2
Shard P3
Data on disk
Node2
Shard R1
Shard R2
Shard R3
Replicas
Node3
API
POST: event
"Event" could be a Nested Document
Time Series Index
Design Choices
March2016, April2016, May2016…
Or
March012016, March022016, March032016…
Or
2016Week1….2016Week52
POST: Event
Store by Month
Store by Day
Store by Week

ElasticSearch Integration with Spark

- ElasticSearh-Hadoop package
  - **elasticsearch-hadoop** allows Elasticsearch to be used in Spark in two ways:
    - through the dedicated support available since 2.1
    - or through the Map/Reduce bridge since 2.0
  - https://www.elastic.co/guide/en/elasticsearch/hadoop/current/spark.html



elasticsearch-hadoop (homepage)

Official integration between Apache Spark and Elasticsearch real-time search and analytics

@elastic / ★★★★★ (3)

Native Java/Scala API for Elasticsearch in Spark. Read/write RDDs and DataFrames from/to Elasticsearch.
Reference documentation available at https://www.elastic.co/guide/en/elasticsearch/hadoop/current/spark.html
Note that artifacts for Scala 2.11 are also available - simply use the _2.11 suffix instead in the artifact id.

Tags

1 sql | 1 elasticsearch | 1 search | 1 analytics | 1 realtime | 1 core | 1 data source

How to [+]

Include this package in your Spark Applications using:

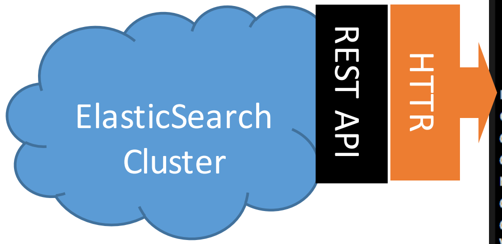
spark-shell, pyspark, or spark-submit

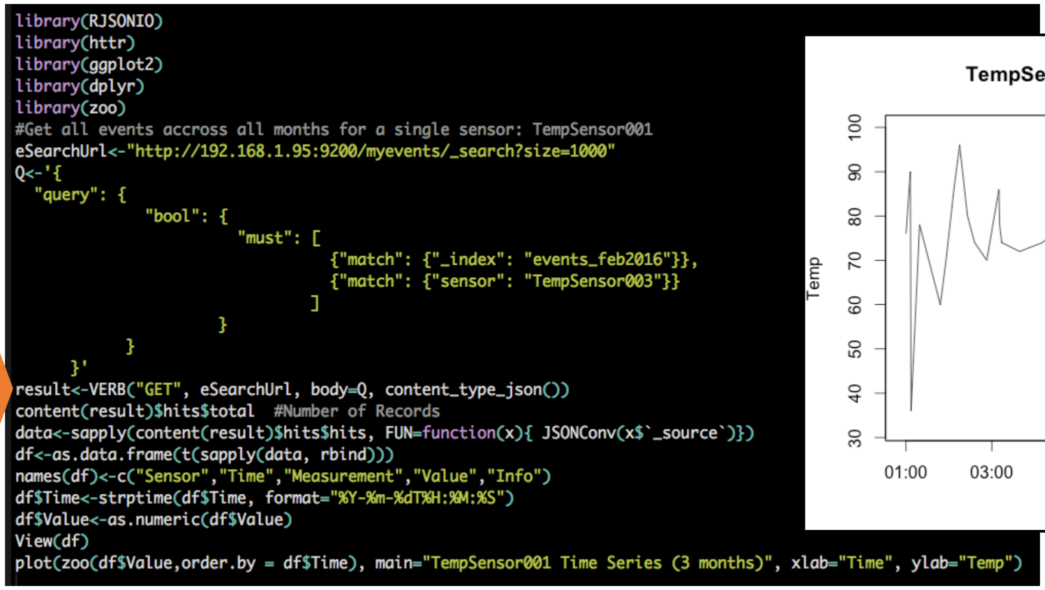
```
> $SPARK_HOME/bin/spark-shell --packages org.elasticsearch:elasticsearch-spark_2.10:2.2.0
```

Releases

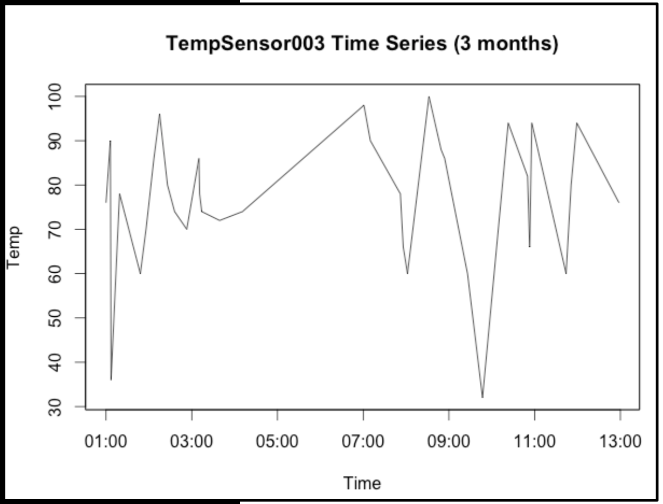
Version: 2.2.0 (c11e37 | zip | jar) / Date: 2016-02-02 / License: Apache-2.0 / Scala version: 2.10

Integration with R





```
library(RJSONIO)
library(httr)
library(ggplot2)
library(dplyr)
library(zoo)
#Get all events accross all months for a single sensor: TempSensor001
eSearchUrl<-"http://192.168.1.95:9200/myevents/_search?size=1000"
Q<-'{
  "query": {
            "bool": {
                      "must": [
                                {"match": {"_index": "events_feb2016"}},
                                {"match": {"sensor": "TempSensor003"}}
                              ]
                    }
          }
      }'
result<-VERB("GET", eSearchUrl, body=Q, content_type_json())
content(result)$hits$total  #Number of Records
data<-sapply(content(result)$hits$hits, FUN=function(x){ JSONConv(x$`_source`)})
df<-as.data.frame(t(sapply(data, rbind)))
names(df)<-c("Sensor","Time","Measurement","Value","Info")
df$Time<-strptime(df$Time, format="%Y-%m-%dT%H:%M:%S")
df$Value<-as.numeric(df$Value)
View(df)
plot(zoo(df$Value,order.by = df$Time), main="TempSensor001 Time Series (3 months)", xlab="Time", ylab="Temp")
```


TempSensor003 Time Series (3 months)
Temp
01:00
03:00
05:00
07:00
09:00
11:00
13:00
Time