elastic{on} 2017

What's the Scoop with ES-Hadoop?

James Baiera and Anoop Sunke

Elastic

2017-03-09

@JimmyThaHat && @apsunke

James Baiera

- Software Engineer @ Elastic
- ES-Hadoop Maintainer
- @*jbaiera* on Github
- @*JimmyThaHat* on Twitter

Agenda

45 minute block

1. Introduction to ES-Hadoop
2. Anatomy of a Job Execution
3. Connector Feature Tour and What's to Come
4. User Success Stories and Use Cases
5. Q&A

Hadoop Overview

In two minutes or less

Hadoop Ecosystem

Hadoop Ecosystem

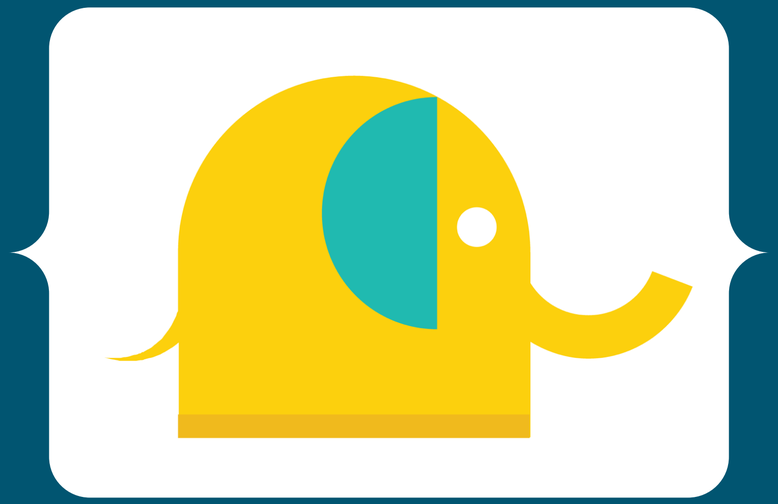
Map / Reduce

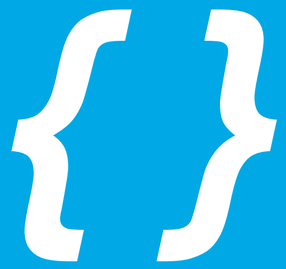
Other

YARN

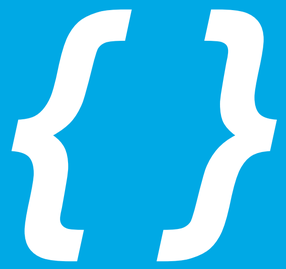
Hadoop Distributed File System (HDFS)

Introduction to ES-Hadoop

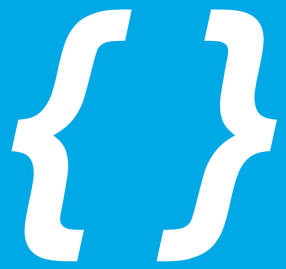




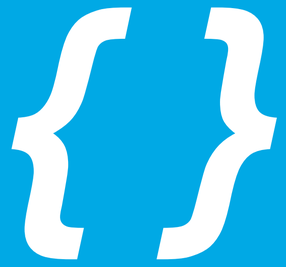
Elasticsearch for Apache Hadoop is an open-source, stand-alone, self-contained, small library that allows Hadoop jobs to interact with Elasticsearch.



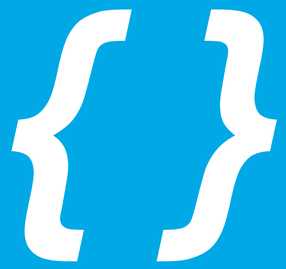
Elasticsearch for Apache Hadoop is an open-source, stand-alone, self-contained, small library that allows Hadoop jobs to interact with Elasticsearch.



Elasticsearch for Apache Hadoop is an open-source, stand-alone, self-contained, small library that allows Hadoop jobs to interact with Elasticsearch.

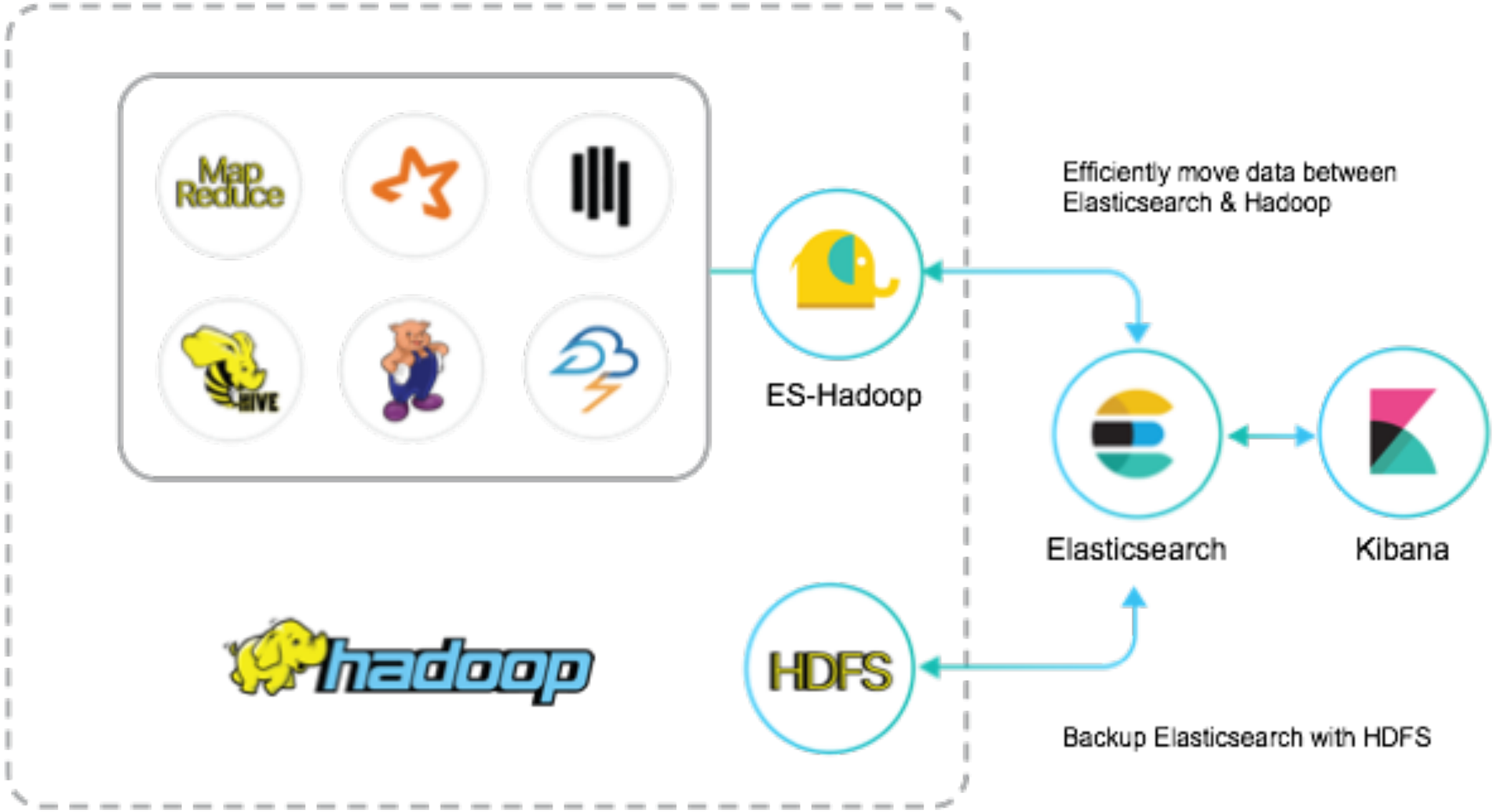


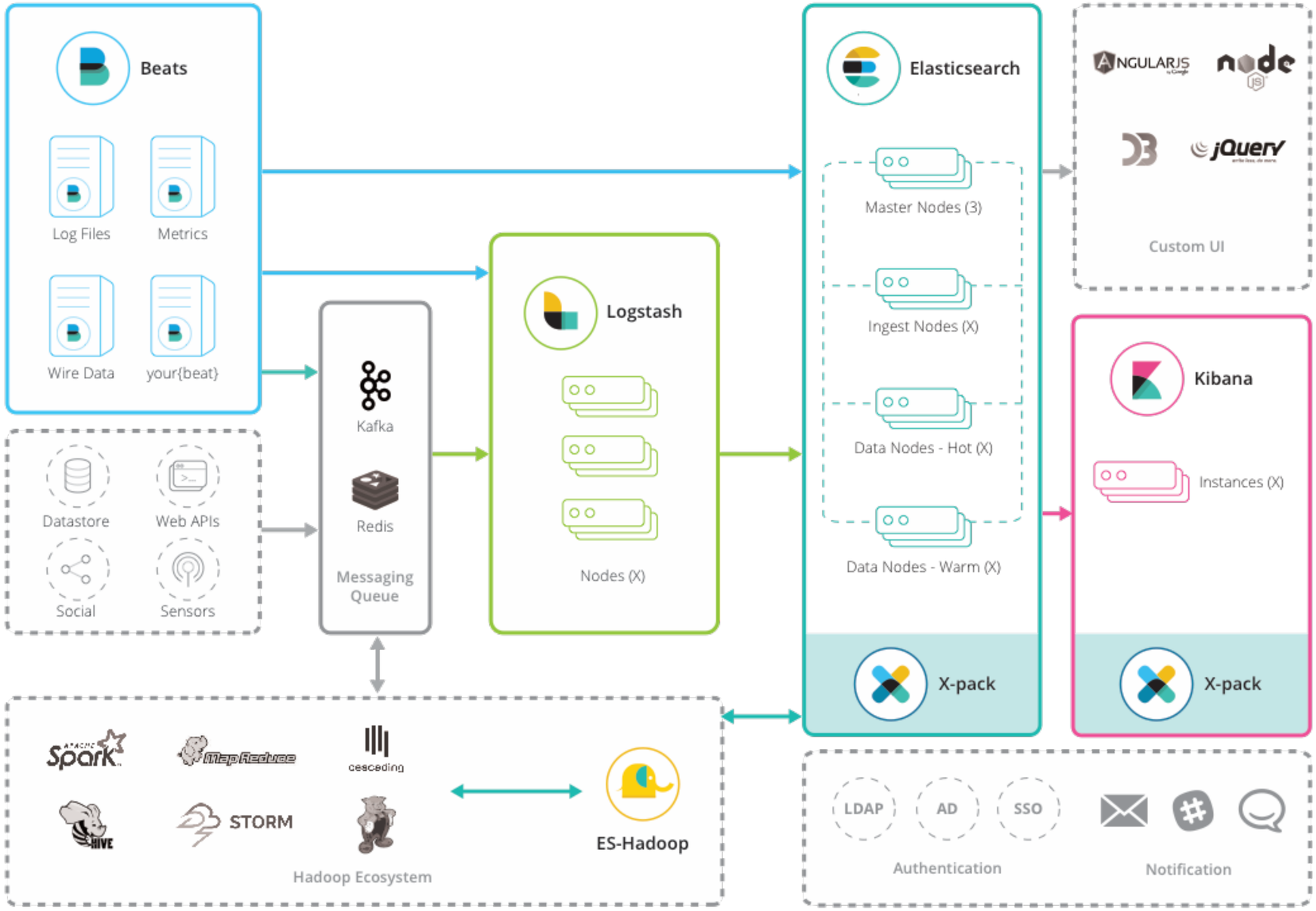
Elasticsearch for Apache Hadoop is an open-source, stand-alone, self-contained, small library that allows Hadoop jobs to interact with Elasticsearch.



Elasticsearch for Apache Hadoop is an open-source, stand-alone, self-contained, small library that allows Hadoop jobs to interact with Elasticsearch.

Birds Eye View




Beats
Log Files
Metrics
Wire Data
your{beat}
Datastore
Web APIs
Social
Sensors
Kafka
Redis
Messaging Queue
Logstash
Nodes (X)
Elasticsearch
Master Nodes (3)
Ingest Nodes (X)
Data Nodes - Hot (X)
Data Nodes - Warm (X)
X-pack
Custom UI
Kibana
Instances (X)
X-pack
Spark
MapReduce
cascading
HIVE
STORM
Hadoop Ecosystem
ES-Hadoop
LDAP
AD
SSO
Authentication
Notification

ES-Hadoop Integrations

Library / API		ES-Hadoop Exposed As
MapReduce		`Input/OutputFormat`
Cascading		`Tap/Sink`
Apache Pig		`Storage (Load and Store)`
Apache Hive		`EXTERNAL Table`
Apache Storm		`Spout/Bolt`
Apache Spark		`RDD, DStream, Dataframe, Dataset, DataSource`

ES-Hadoop Features

- Latest Spark 2.1 support
- Adaptive I/O for error handling, re-routing, backpressure
- Push-down processing on either platform
- Co-location, rack awareness
- Elastic security compatible - basic authentication, SSL/TLS, PKI
- Hadoop Kerberos security compatible
- Hadoop distribution agnostic

Why?

Peeling Back the Layers

ES-Hadoop Functional Deep Dive

Pushdown

Projections and Predicates

ID : Long	First Name : String	Age : Integer	Profile : Text
347	Martha	32	Has a fancy vineyard
348	Mary	37	Friends with Peter and Paul
349	Geoff	25	Spells name with a ‘G’
350	Travis	23	Loves jazz music
351	Jeremiah	39	Bullfrog, Good Friend
352	Mark	42	Hates cold spaghetti

Predicates

Predicate: ID > 347 && ID <= 350

ID : Long	First Name : String	Age : Integer	Profile : Text
347	Martha	32	Has a fancy vineyard
348	Mary	37	Friends with Peter and Paul
349	Geoff	25	Spells name with a ‘G’
350	Travis	23	Loves jazz music
351	Jeremiah	39	Bullfrog, Good Friend
352	Mark	42	Hates cold spaghetti

Projections

Projection: Select (ID, FirstName, Age)

ID : Long	FirstName : String	Age : Integer	Profile : Text
347	Martha	32	Has a fancy vineyard
348	Mary	37	Friends with Peter and Paul
349	Geoff	25	Spells name with a ‘G’
350	Travis	23	Loves jazz music
351	Jeremiah	39	Bullfrog, Good Friend
352	Mark	42	Hates cold spaghetti

Final View

Select (ID, FirstName, Age) where ID > 347 && ID <= 350

ID : Long	First Name : String	Age : Integer	Profile : Text
347	Martha	32	Has a fancy vineyard
348	Mary	37	Friends with Peter and Paul
349	Geoff	25	Spells name with a ‘G’
350	Travis	23	Loves jazz music
351	Jeremiah	39	Bullfrog, Good Friend
352	Mark	42	Hates cold spaghetti

Spark SQL Reading

Predicate Pushdown

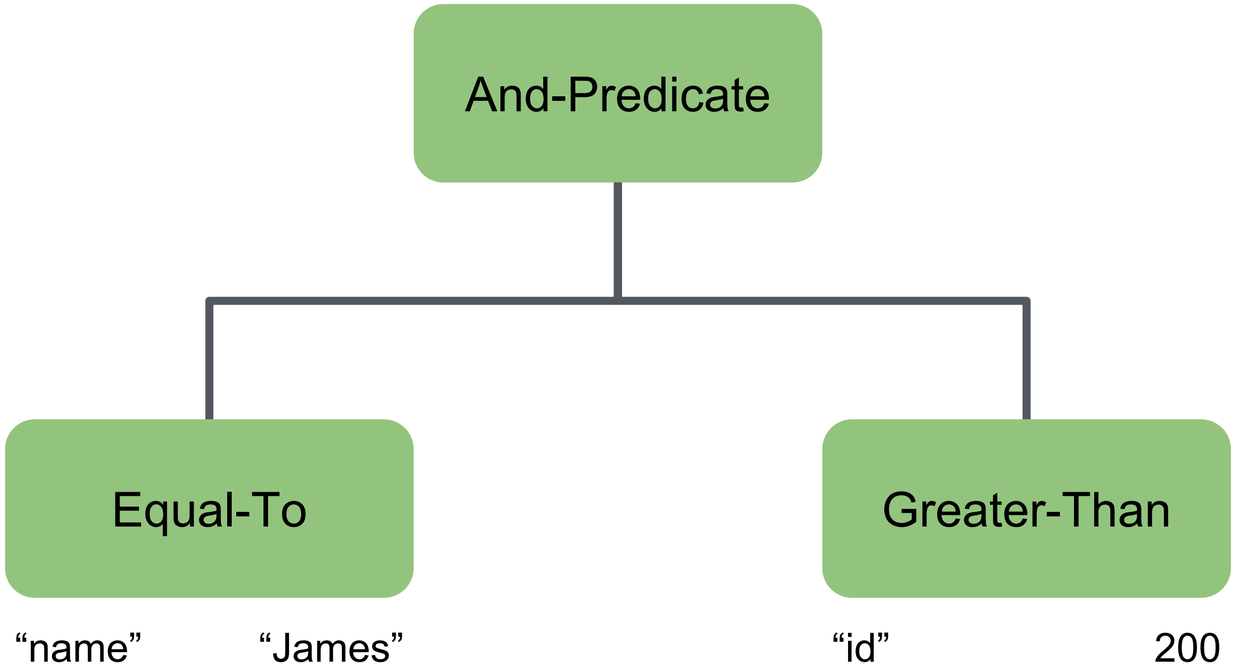
```
val df = sqlContext.read().format("es").load("spark/users")

df.printSchema()
// root
//|-- name: string (nullable = true)
//|-- id: long (nullable = true)
//|-- profile: string (nullable = true)

val filter = df.filter(
    df("name").equalTo("James").and(df("id").gt(200)))
```

Spark SQL Reading

Predicate Pushdown



Spark SQL Reading

Predicate Pushdown

```
{
  "query" : {
    "bool" : {
      "must" : [ { "match_all" : {}} ],
      "filter" : [ {
        "bool":{"filter":[
          {"match" : { "name" : "James"}},
          {"range" : { "id" : { "gt" : 200 }}}
        ]
      }
    }
  }
}
```

Spark SQL Reading

Projection Pushdown

```
val df = sqlContext.read().format("es").load("spark/users")
df.createOrReplaceTempView(“myIndex”)

df.printSchema()
// root
//|-- name: string (nullable = true)
//|-- id: long (nullable = true)
//|-- profile: string (nullable = true)

val names = sqlContext.sql("SELECT name, id FROM myIndex")
```

Spark SQL Reading

Projection Pushdown

```
{
  "_source": [ "name", "id" ],
  "query" : {
    "bool":{
      "must" : [ { "match_all" : {}} ]
    }
  }
}
```

Spark SQL Reading

Projection Pushdown + Predicate Pushdown

```
val df = sqlContext.read().format("es").load("spark/users")
df.createOrReplaceTempView(“myIndex”)

df.printSchema()
// root
//|-- name: string (nullable = true)
//|-- id: long (nullable = true)
//|-- profile: string (nullable = true)

val names = df.sqlContext.sql(
    "SELECT name FROM myIndex WHERE id >=1 AND id <= 10")
```

Spark SQL Reading

Projection Pushdown + Predicate Pushdown

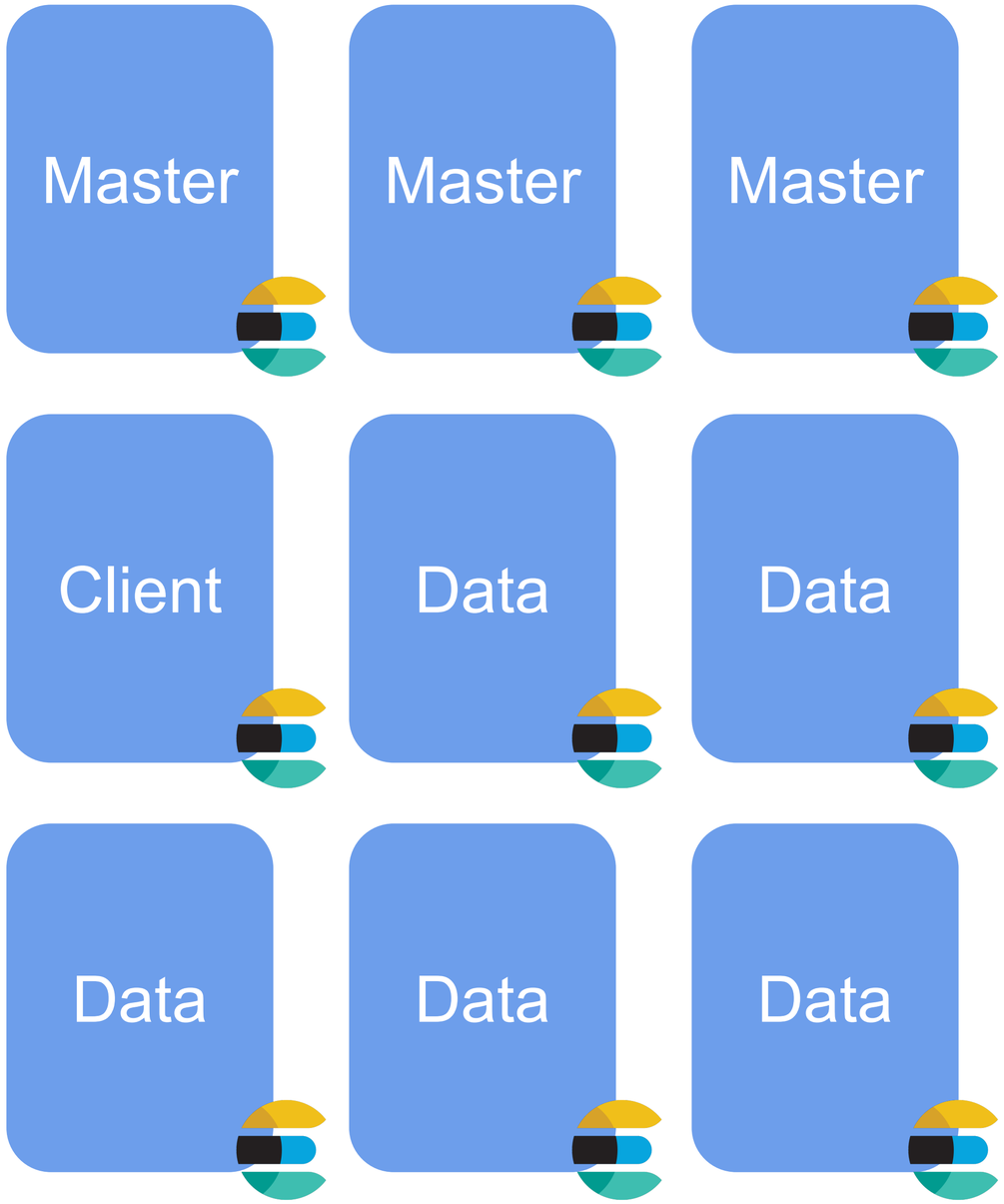
```
{
  "_source": [ "name", "id" ],
  "query" : {
    "bool":{
      "must" : [ { "match_all" : {}} ],
      "filter" : [ {
        "bool":{"filter":[
          { "range" : { "id" : { "gte" : 1 }}},
          { "range" : { "id" : { "lte" : 10 }}}
        ]}
      }]
    }
  }
}
```

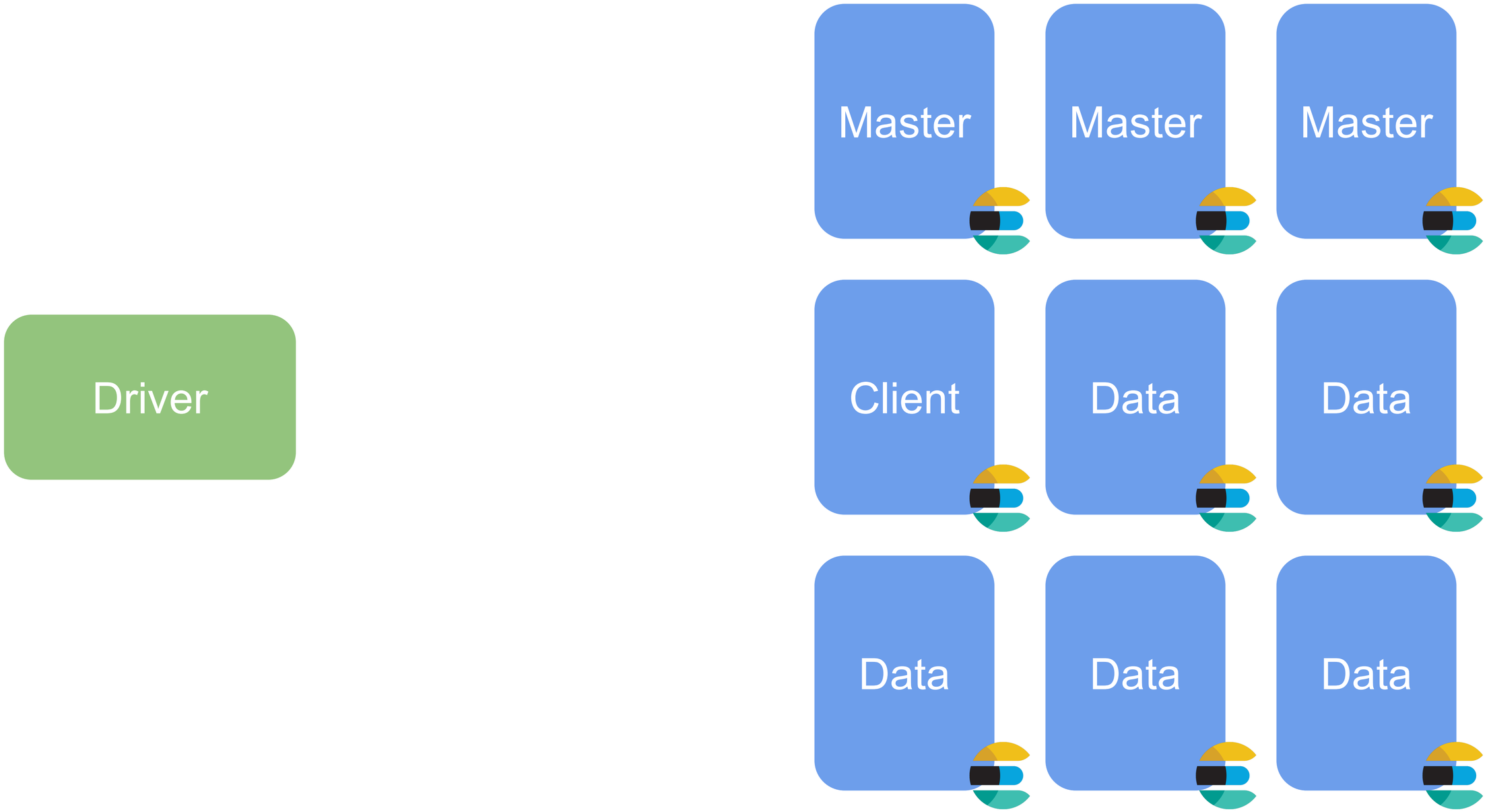
Automatic Pushdown Support in ES-Hadoop

Library / API	Projection	Predicate
MapReduce 	Manual	Manual
Cascading 	Automatic	Manual
Apache Pig 	Automatic	Manual
Apache Hive 	Automatic	Manual
Apache Storm 	Manual	Manual
Apache Spark 	(SQL) Automatic (RDD) Manual	(SQL) Automatic (RDD) Manual

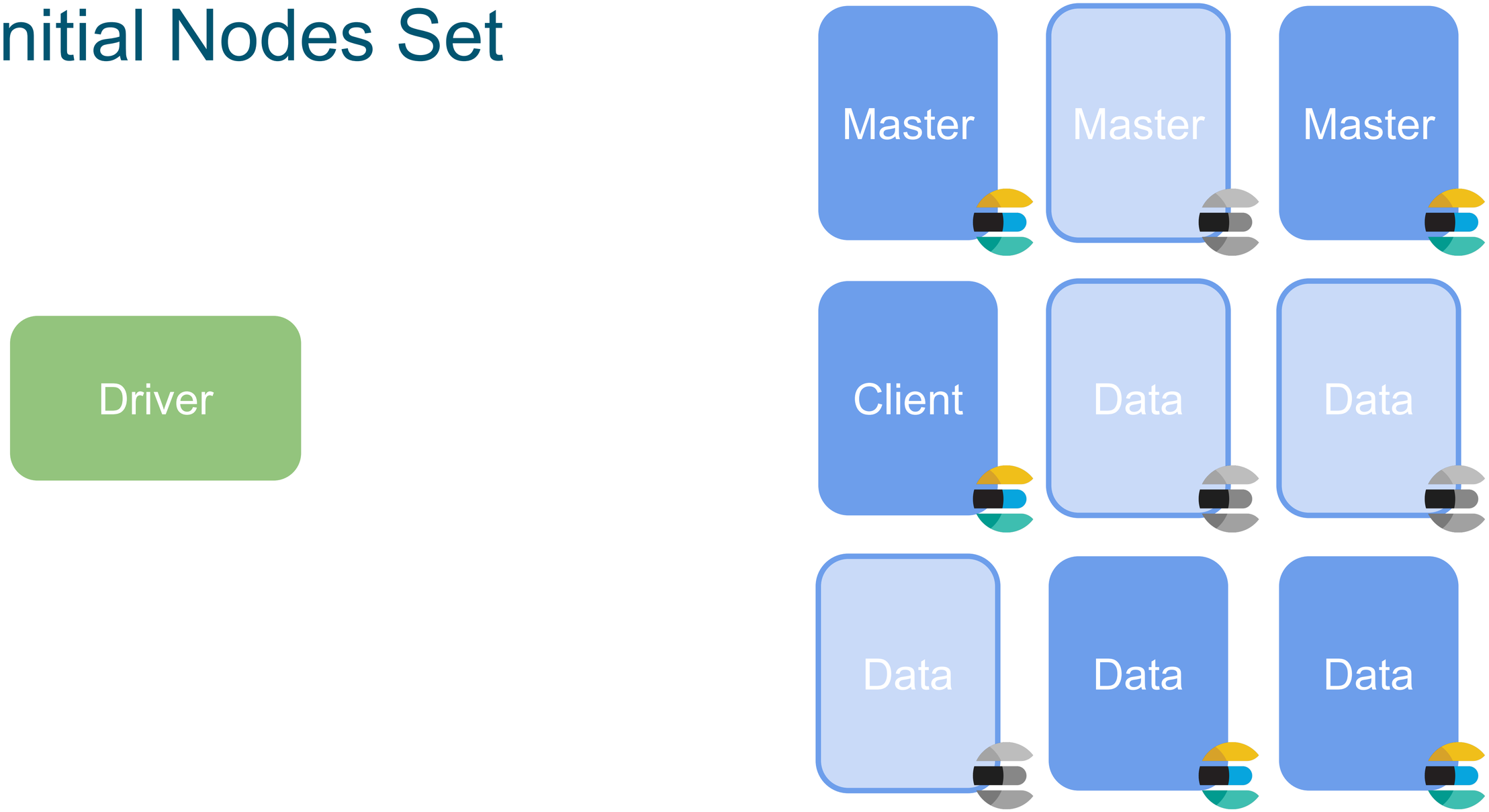
Node Discovery


elasticsearch

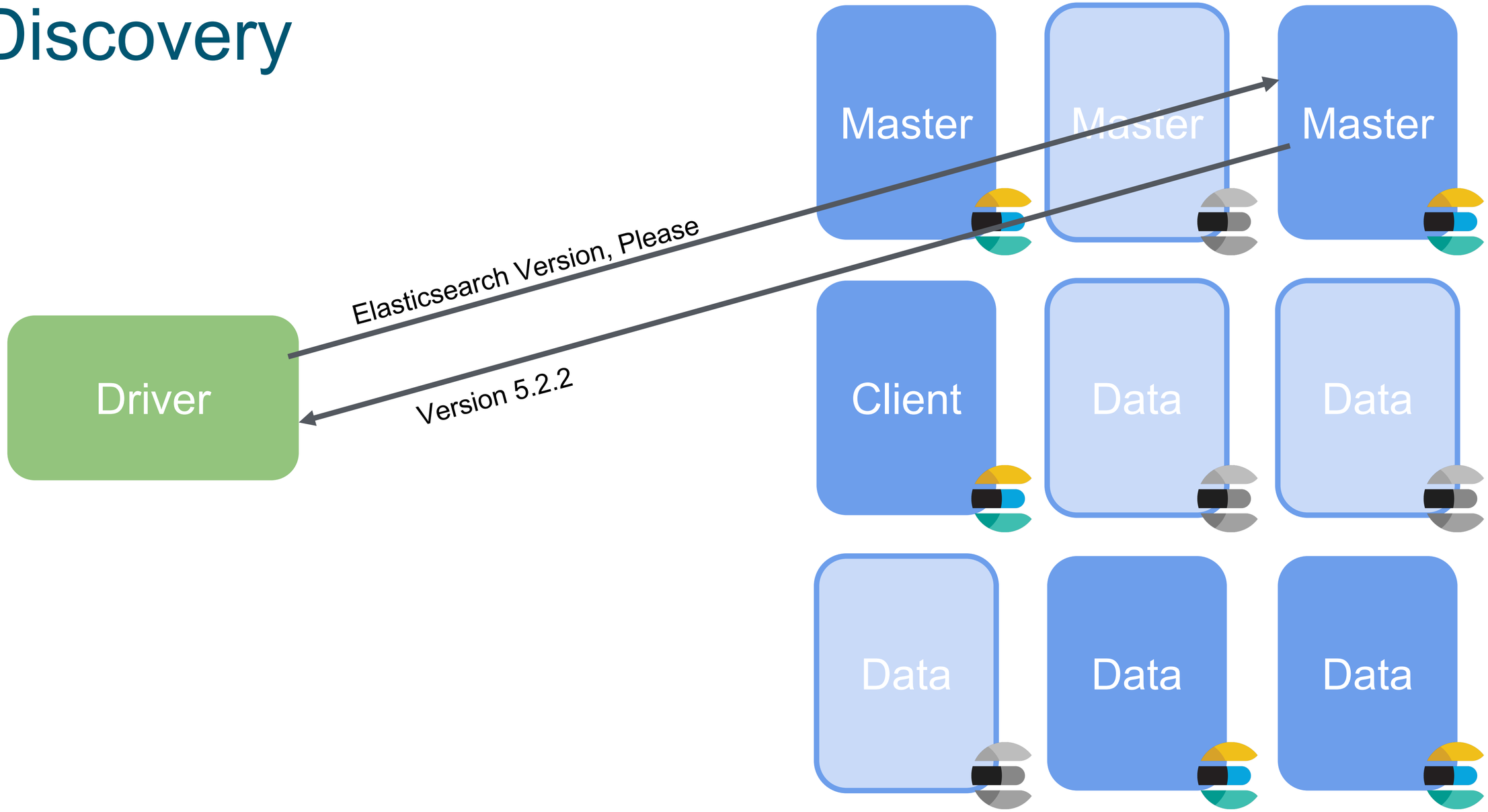

Master
Master
Master
Client
Data
Data
Data
Data
Data


Master
Master
Master
Client
Data
Data
Data
Data
Data
Driver

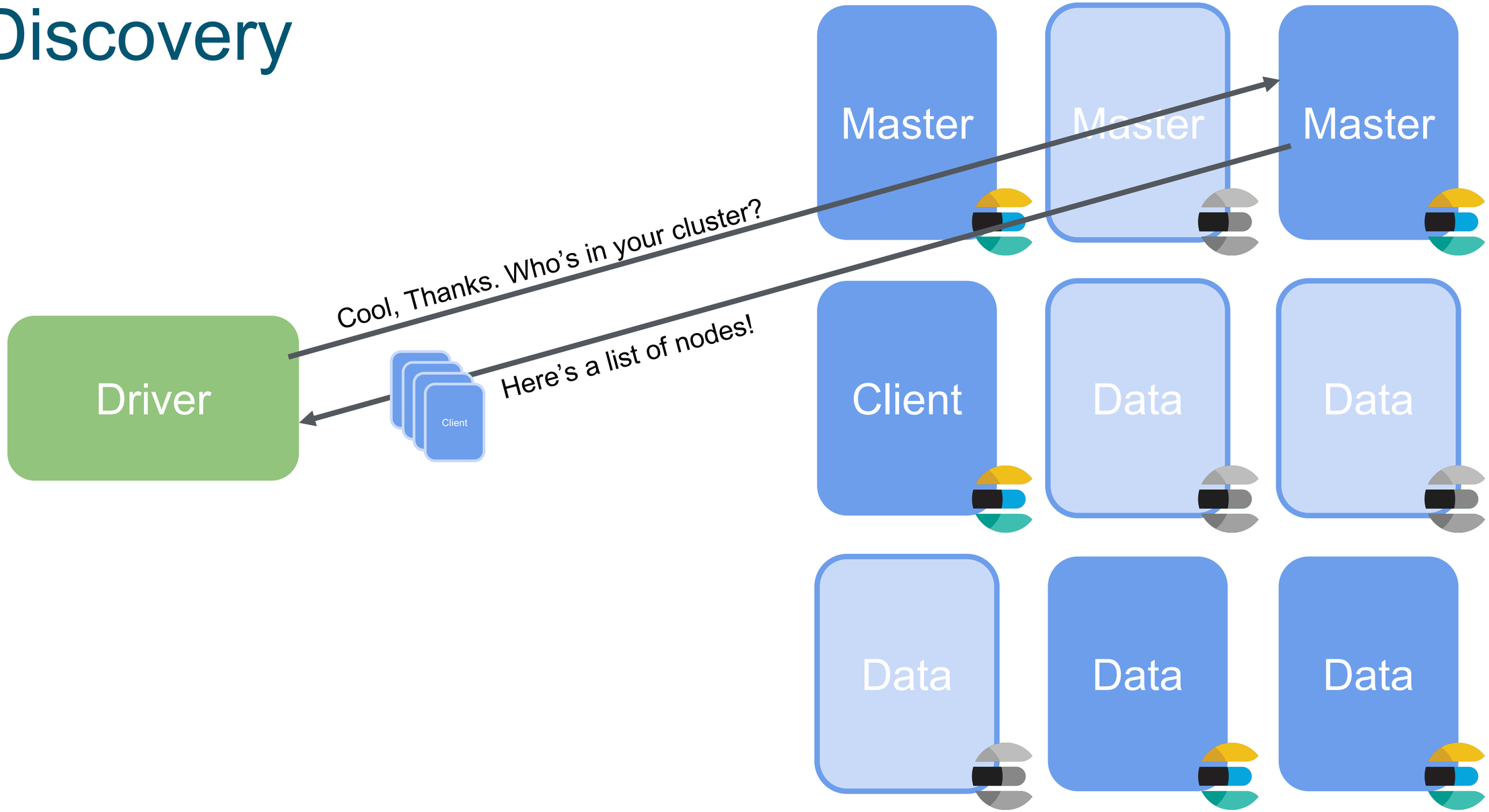
Initial Nodes Set



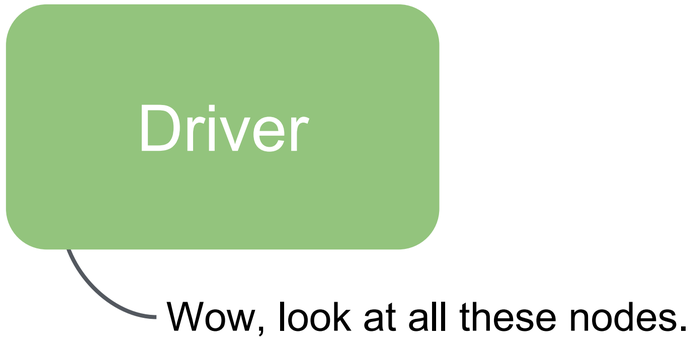
Discovery

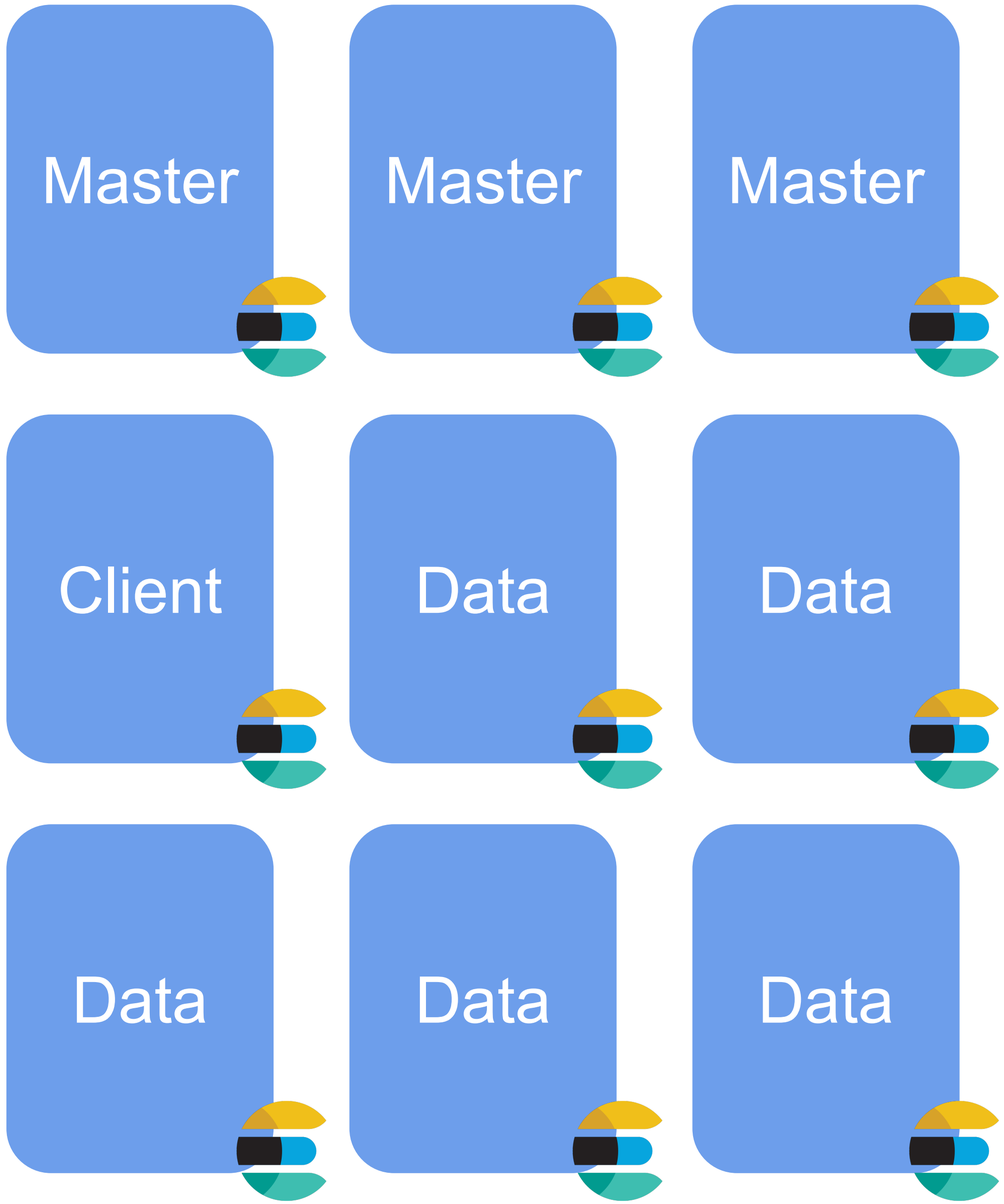

Master
Master
Master
Elasticsearch Version, Please
Version 5.2.2
Driver
Client
Data
Data
Data
Data
Data

Discovery

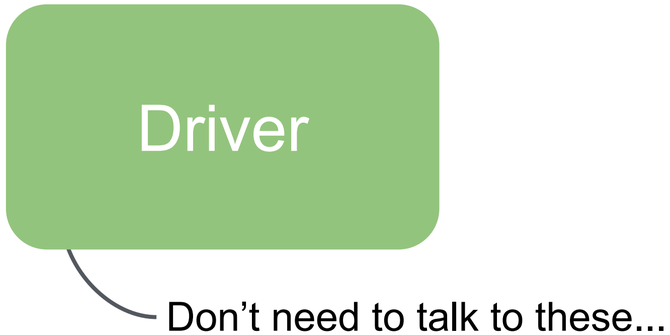

Master
Master
Master
Cool, Thanks. Who's in your cluster?
Here's a list of nodes!
Driver
Client
Client
Data
Data
Data
Data
Data

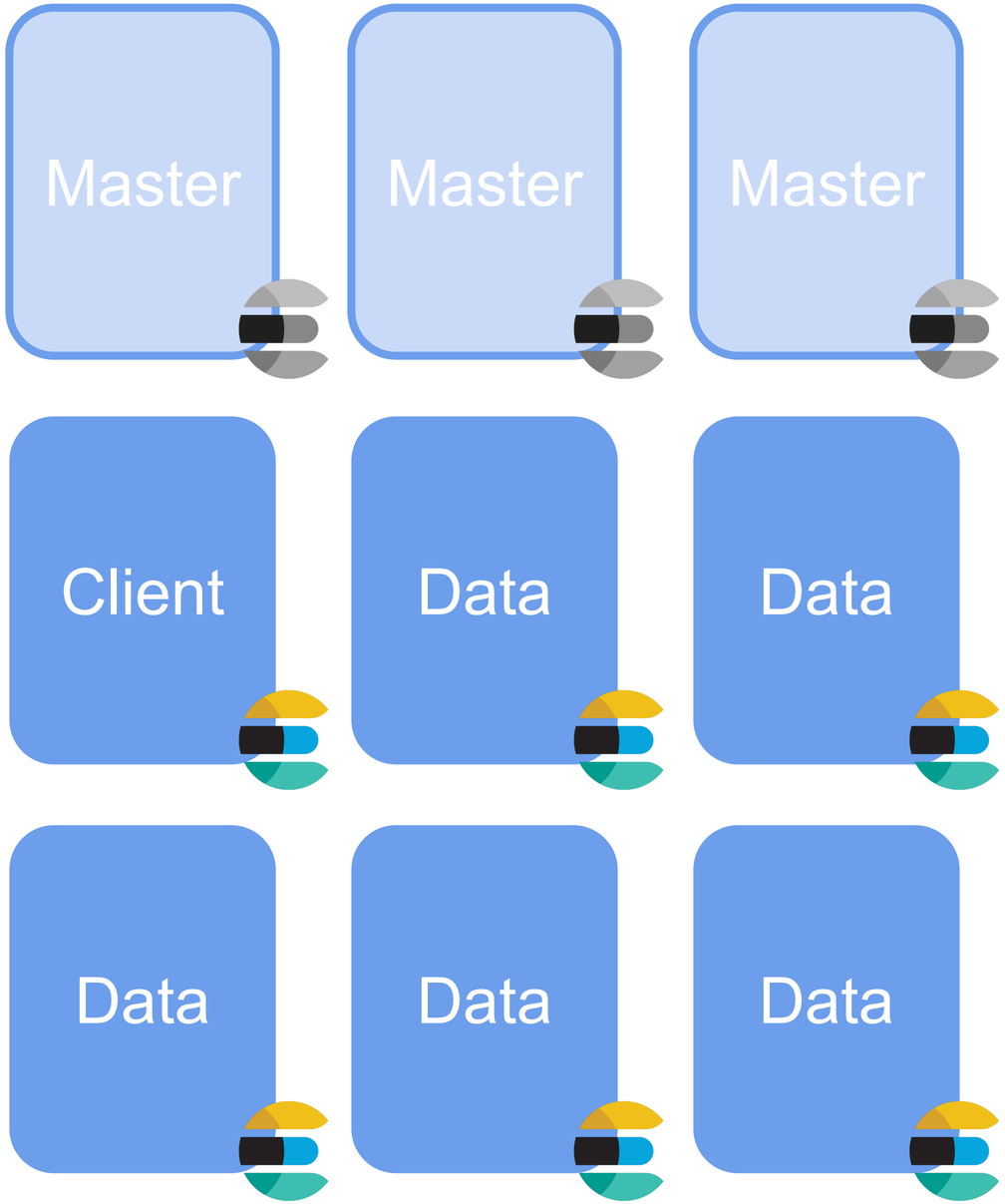
Discovery


Driver
Wow, look at all these nodes.

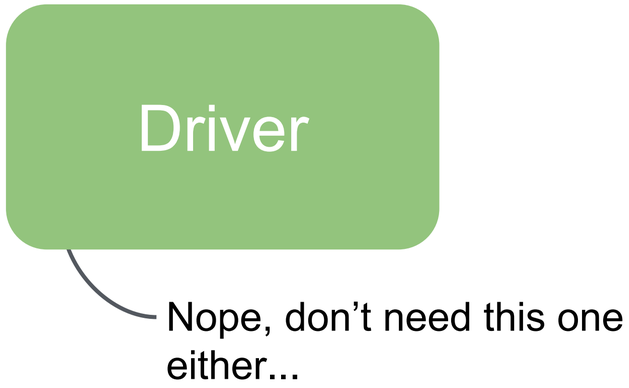

Master
Master
Master
Client
Data
Data
Data
Data
Data

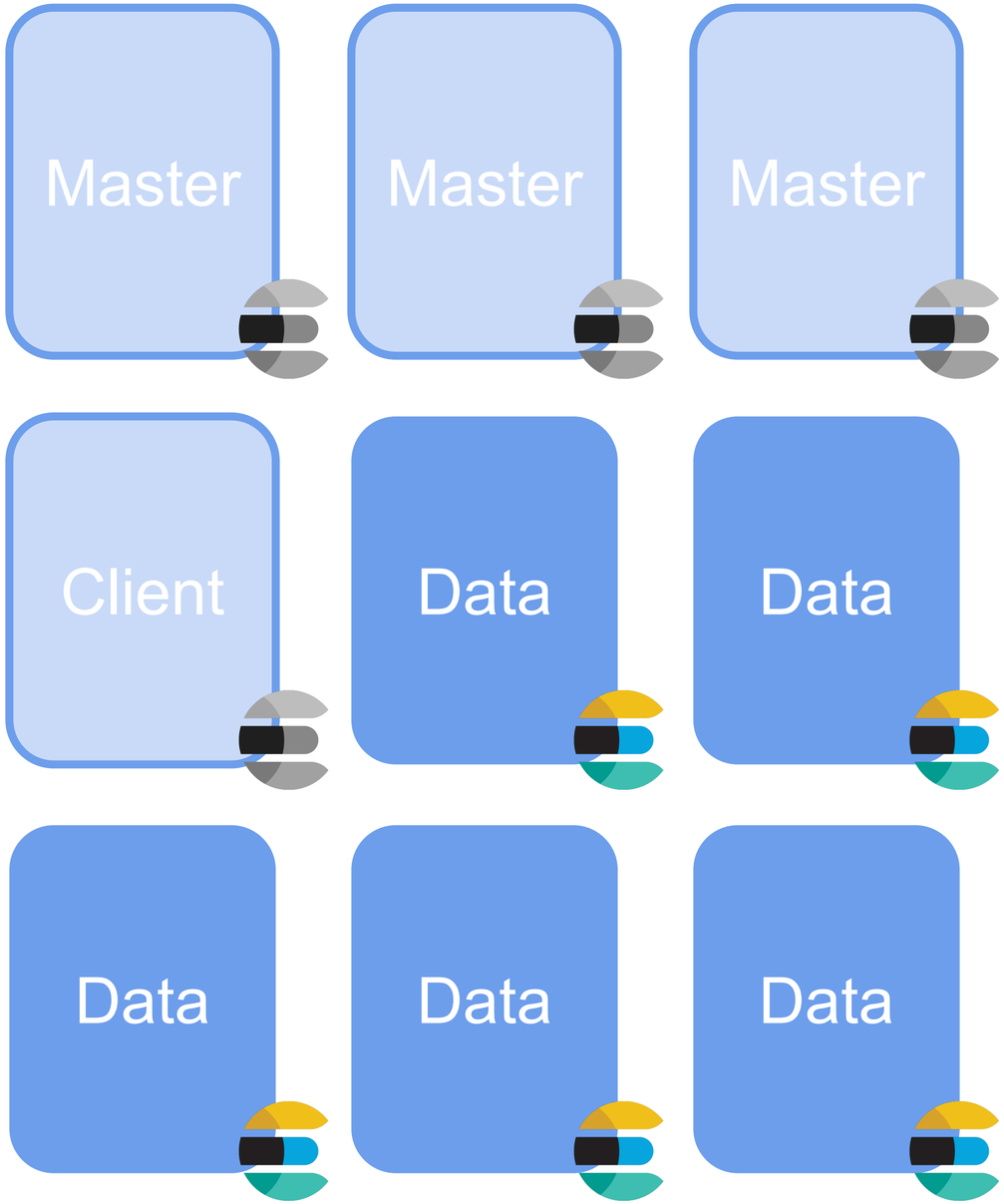
Discovery


Driver
Don’t need to talk to these...

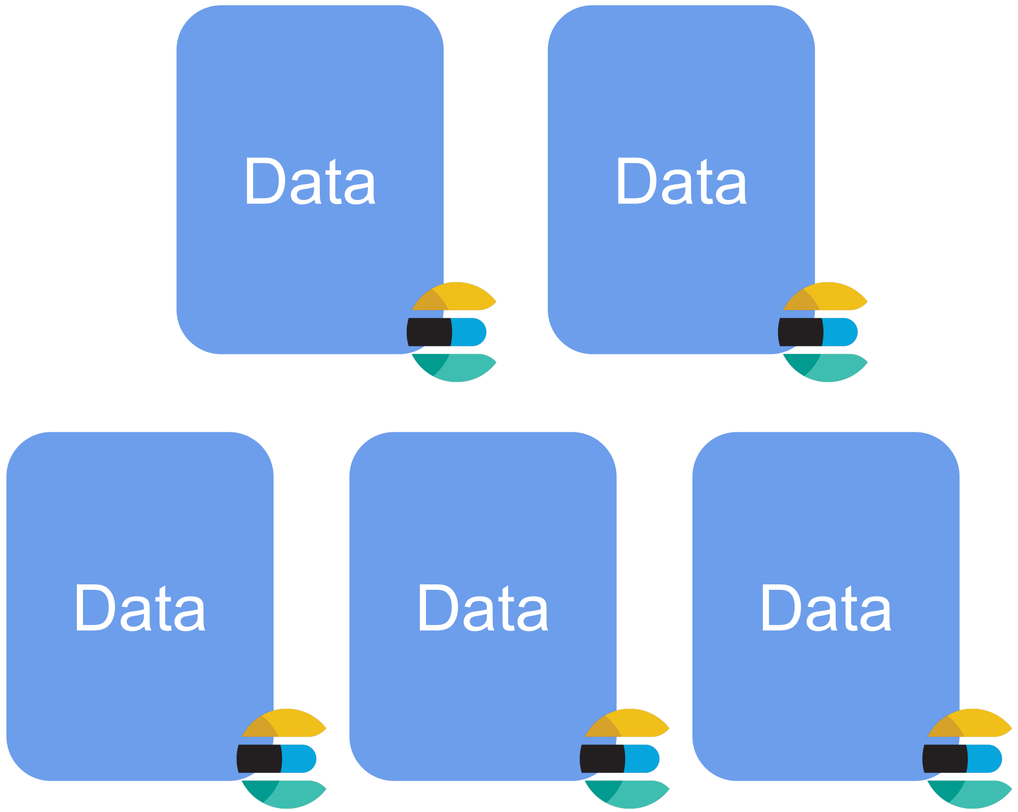

Master
Master
Master
Client
Data
Data
Data
Data
Data

Discovery


Driver
Nope, don’t need this one either...


Master
Master
Master
Client
Data
Data
Data
Data
Data

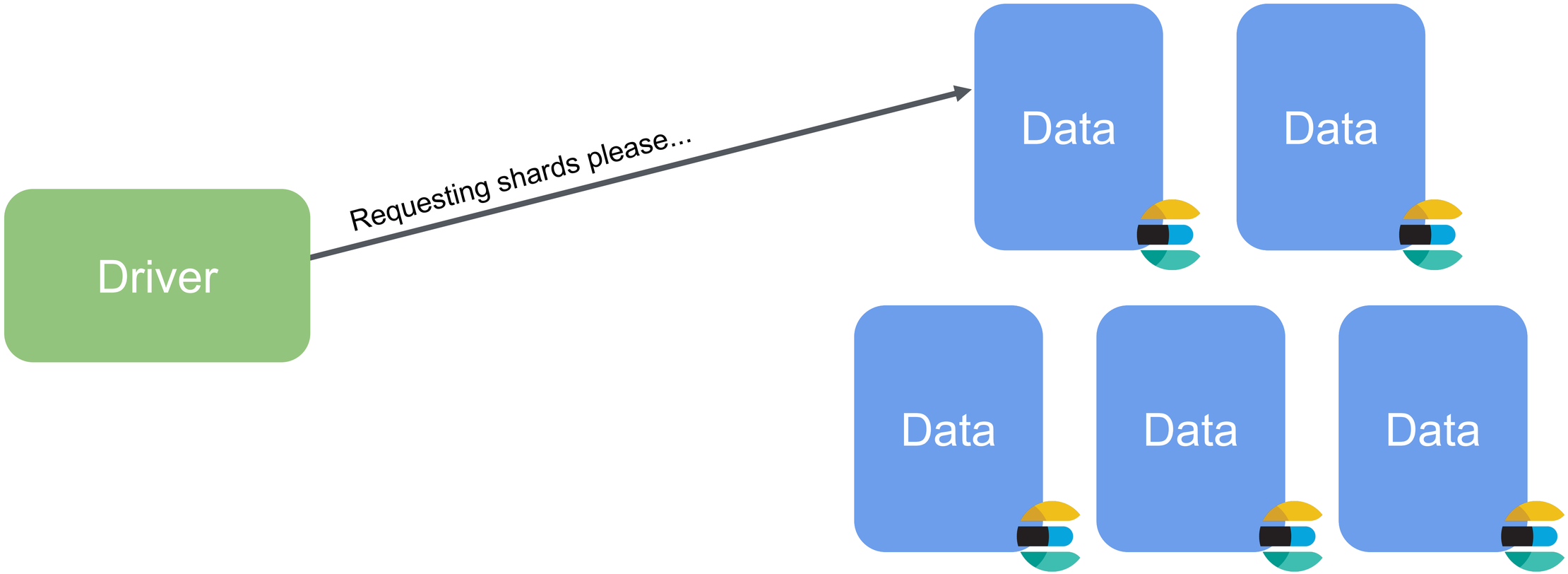
Driver



Reading from Elasticsearch

Finding Partitions

Reading index “logs/data”...



Finding Partitions

Reading index “logs/data”...

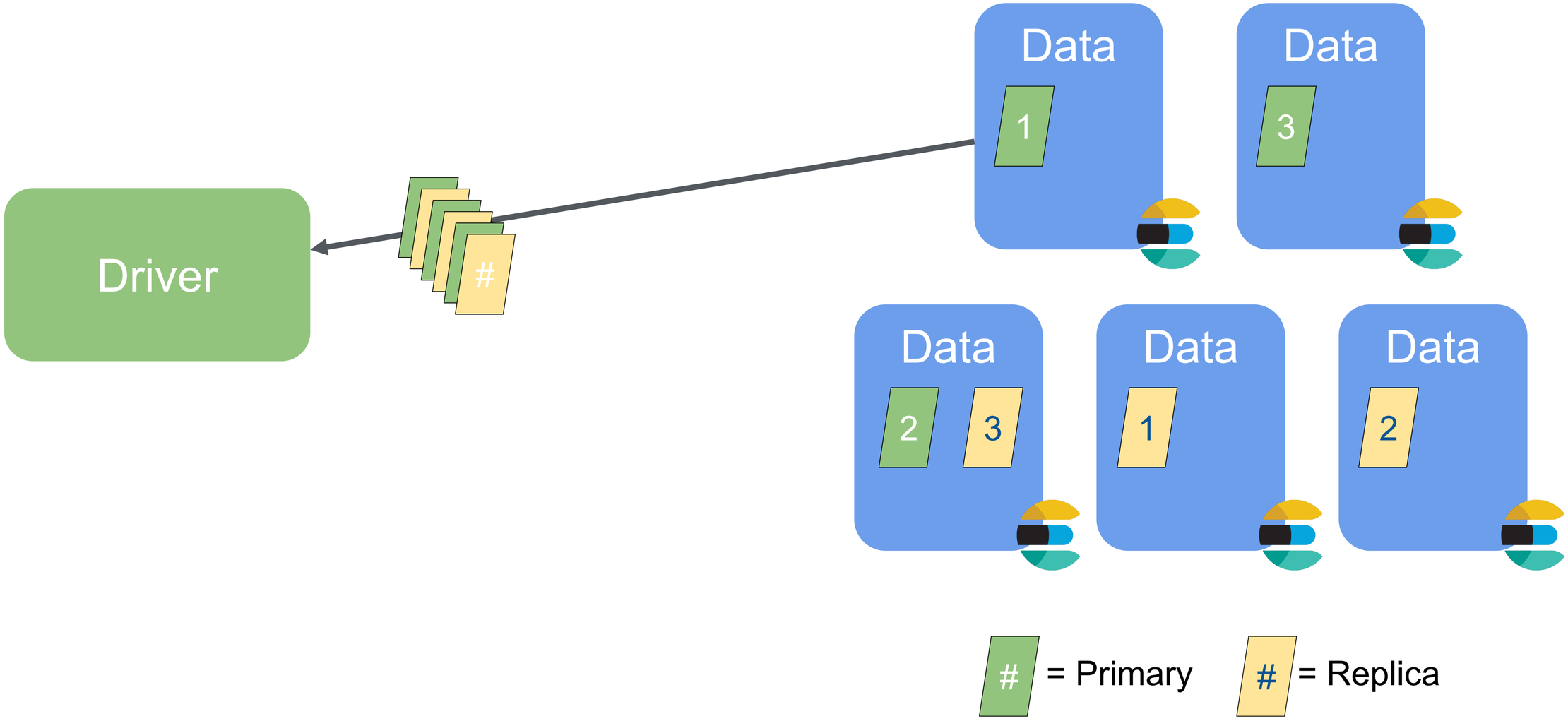
Driver

= Primary

= Replica

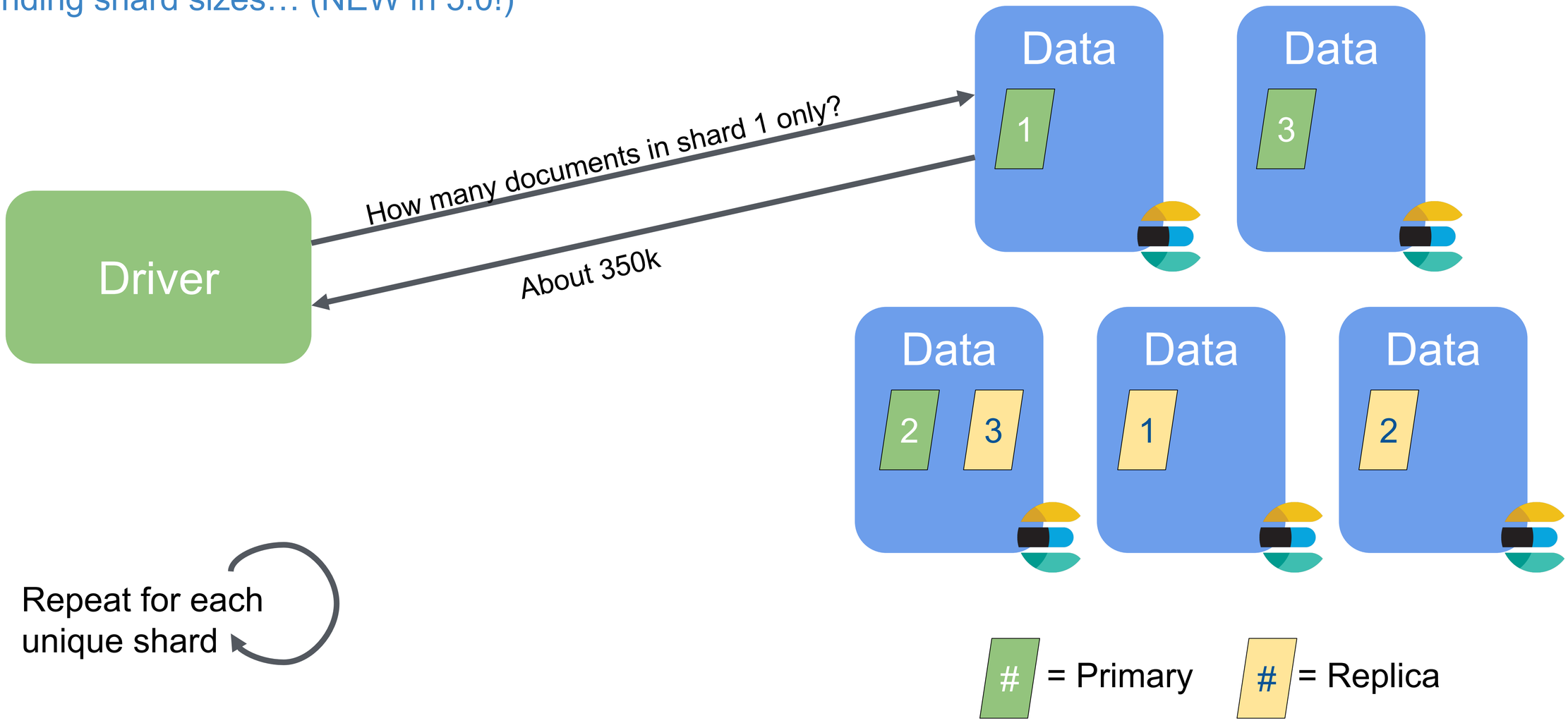
Finding Partitions

Reading index “logs/data”...



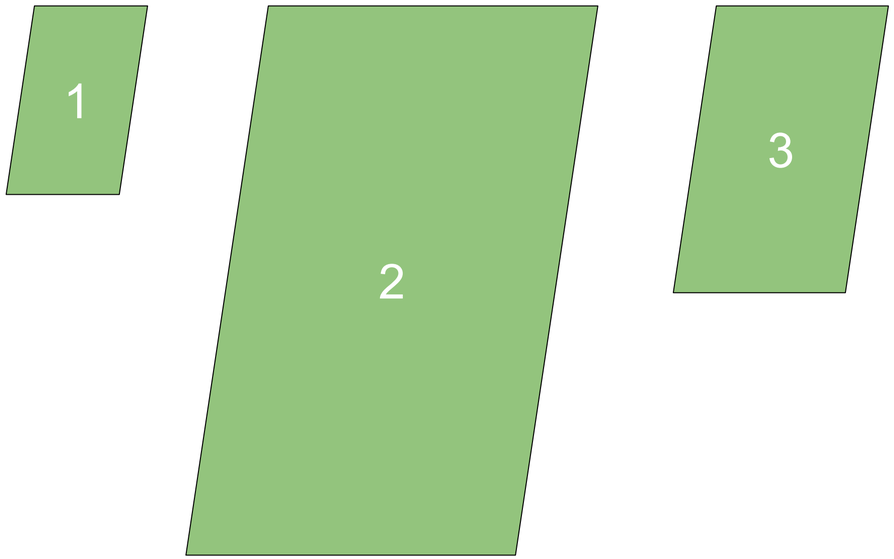
Finding Partitions

Reading index “logs/data”...
Finding shard sizes… (NEW in 5.0!)



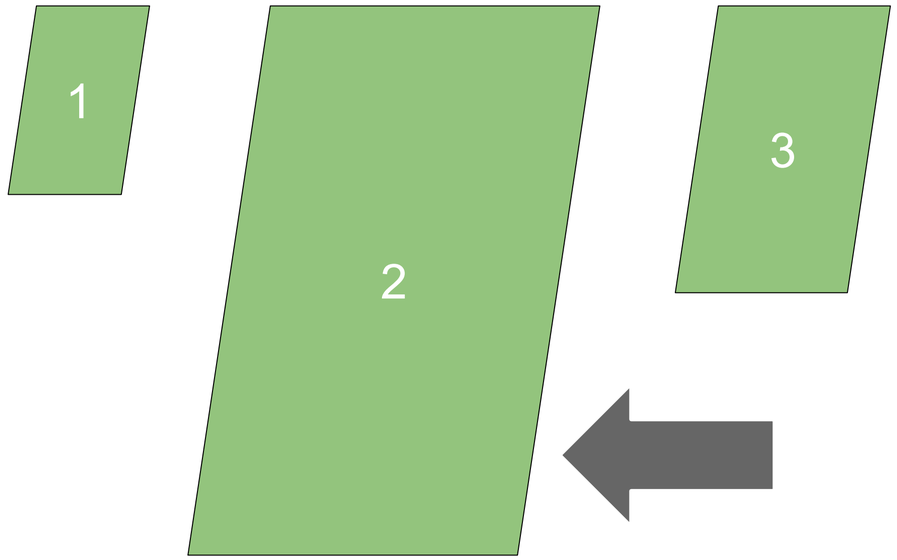
Finding Partitions

Reading index “logs/data”...
Subdividing shards… (NEW in 5.0!)



Finding Partitions

Reading index “logs/data”...
Subdividing shards… (NEW in 5.0!)



NEW IN 5.0 OF ELASTICSEARCH

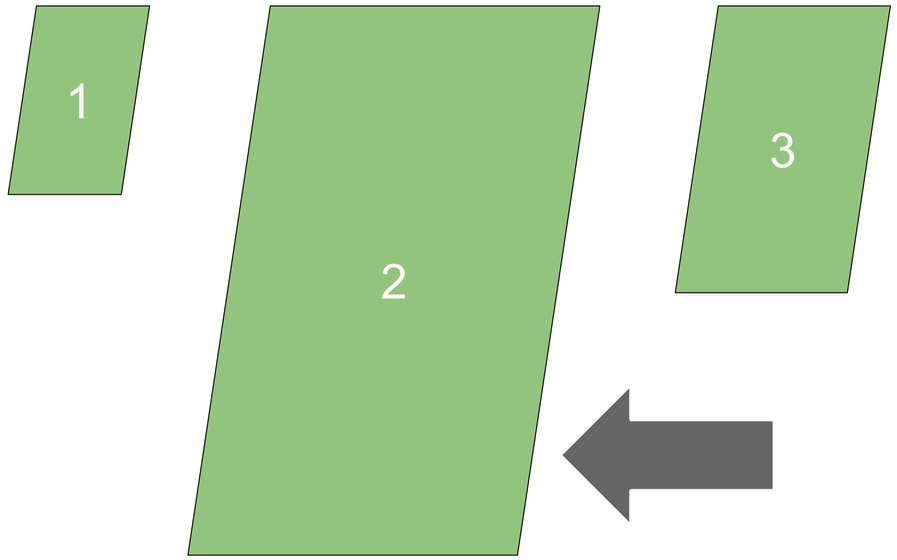
Sliced Scrolls

New in Elasticsearch 5.0

```
curl -XGET localhost:9200/idx/t/_search?scroll=1m -d' {
    "slice": {
        "id": 0,
        "max": 2
    },
    "query": {
        "match" : {
            "title" : "elasticsearch"
        }
    }
}'
```

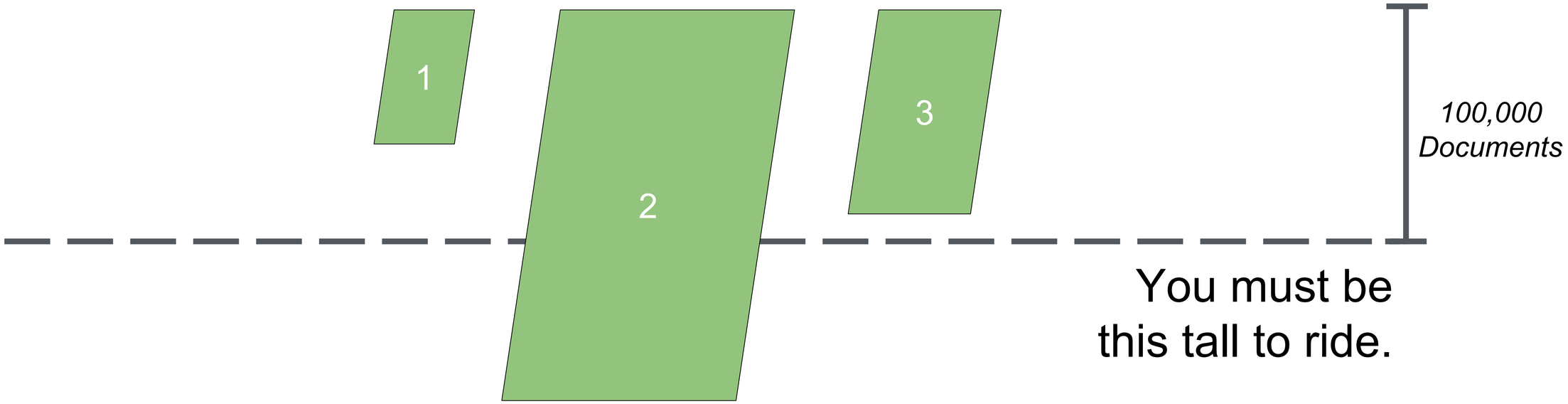
Finding Partitions

Reading index “logs/data”...
Subdividing shards… (NEW in 5.0!)



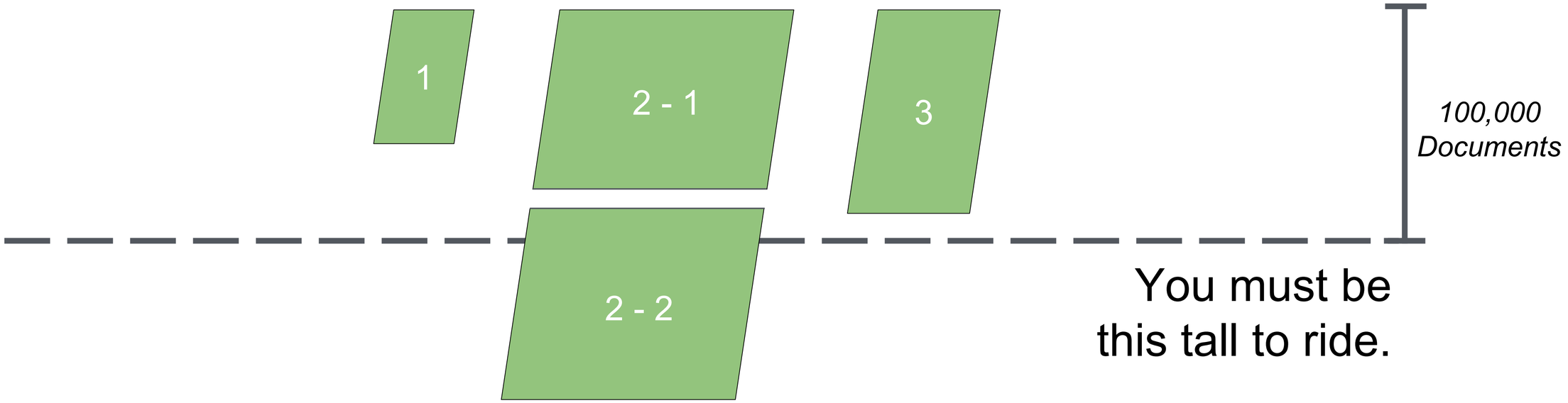
Finding Partitions

Reading index “logs/data”...
Subdividing shards… (NEW in 5.0!)



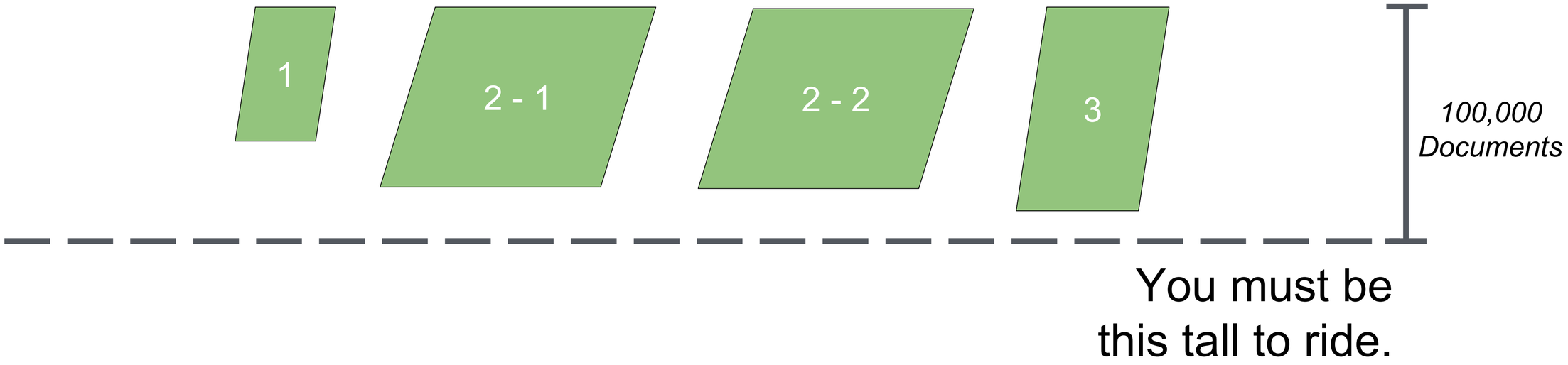
Finding Partitions

Reading index “logs/data”...
Subdividing shards… (NEW in 5.0!)



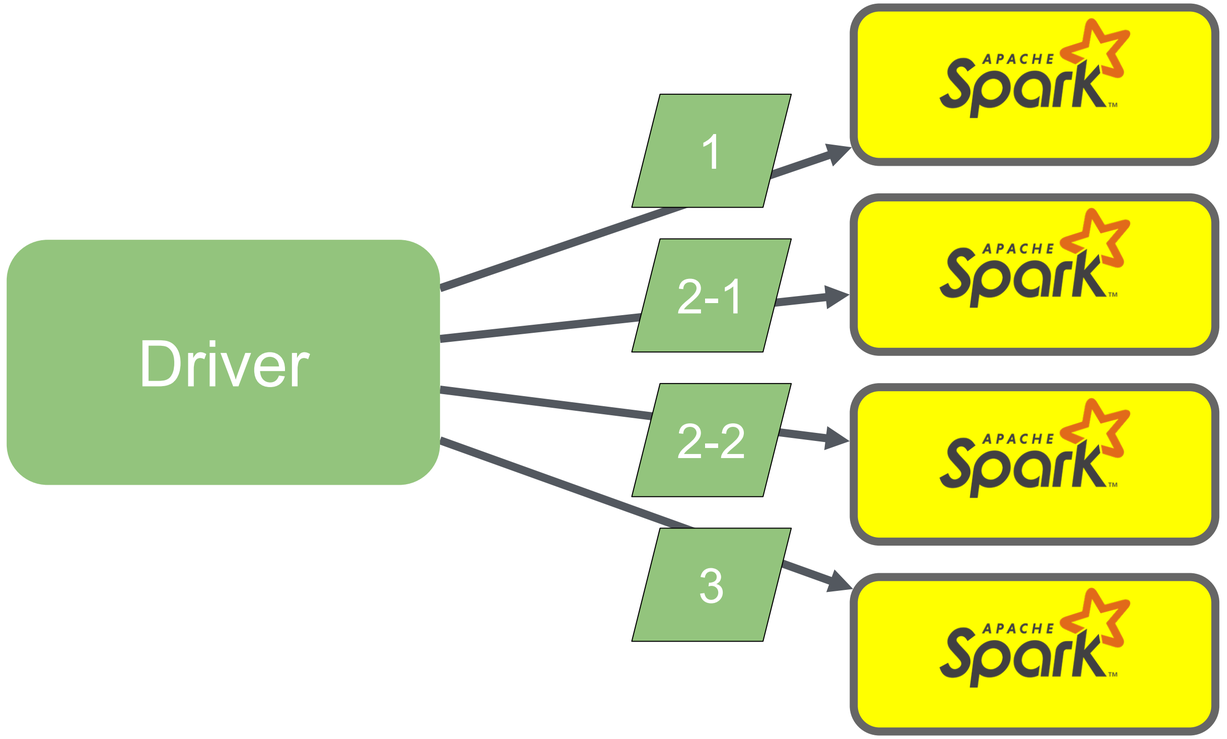
Finding Partitions

Reading index “logs/data”...
Subdividing shards… (NEW in 5.0!)

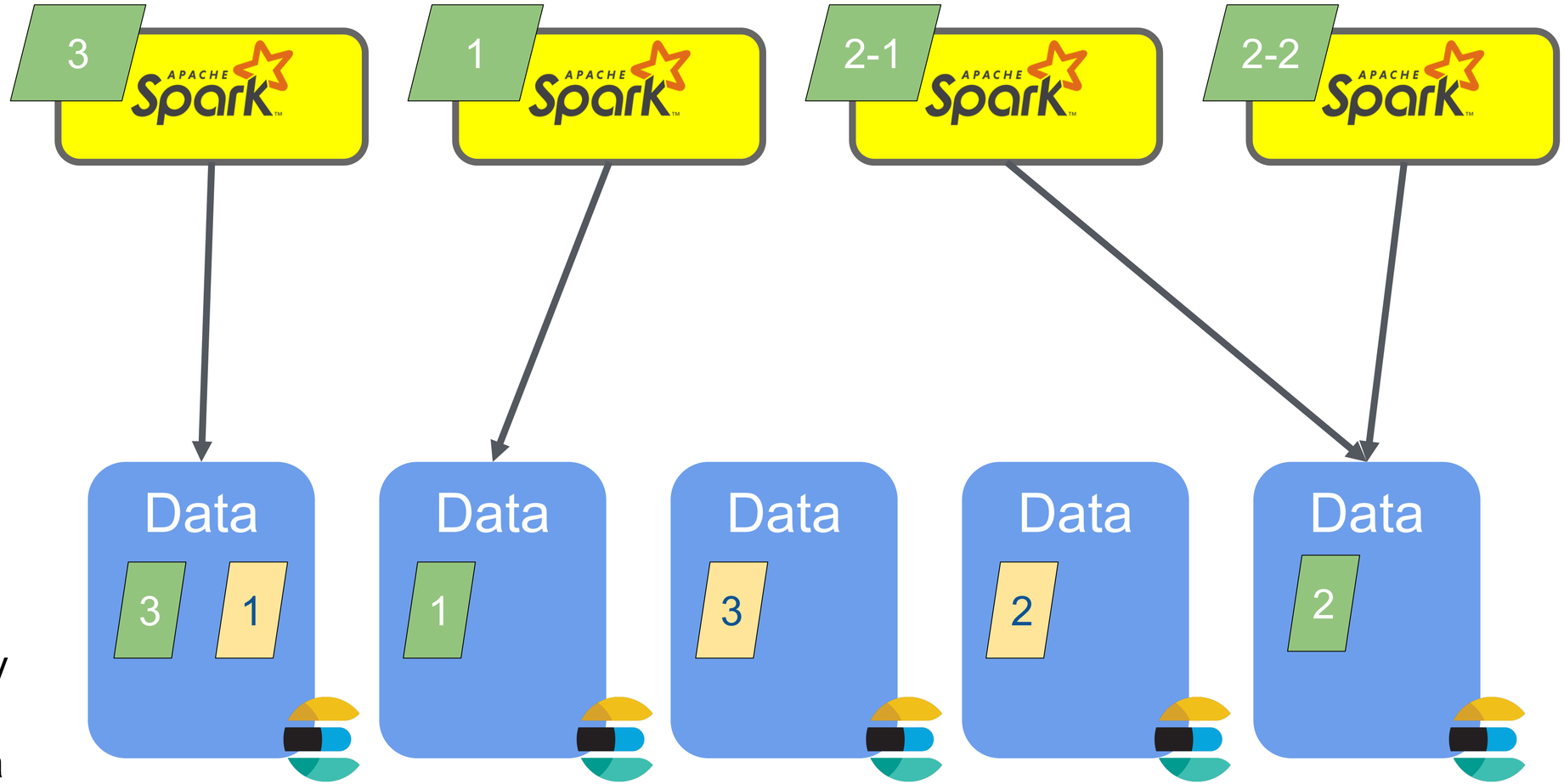


Job Execution

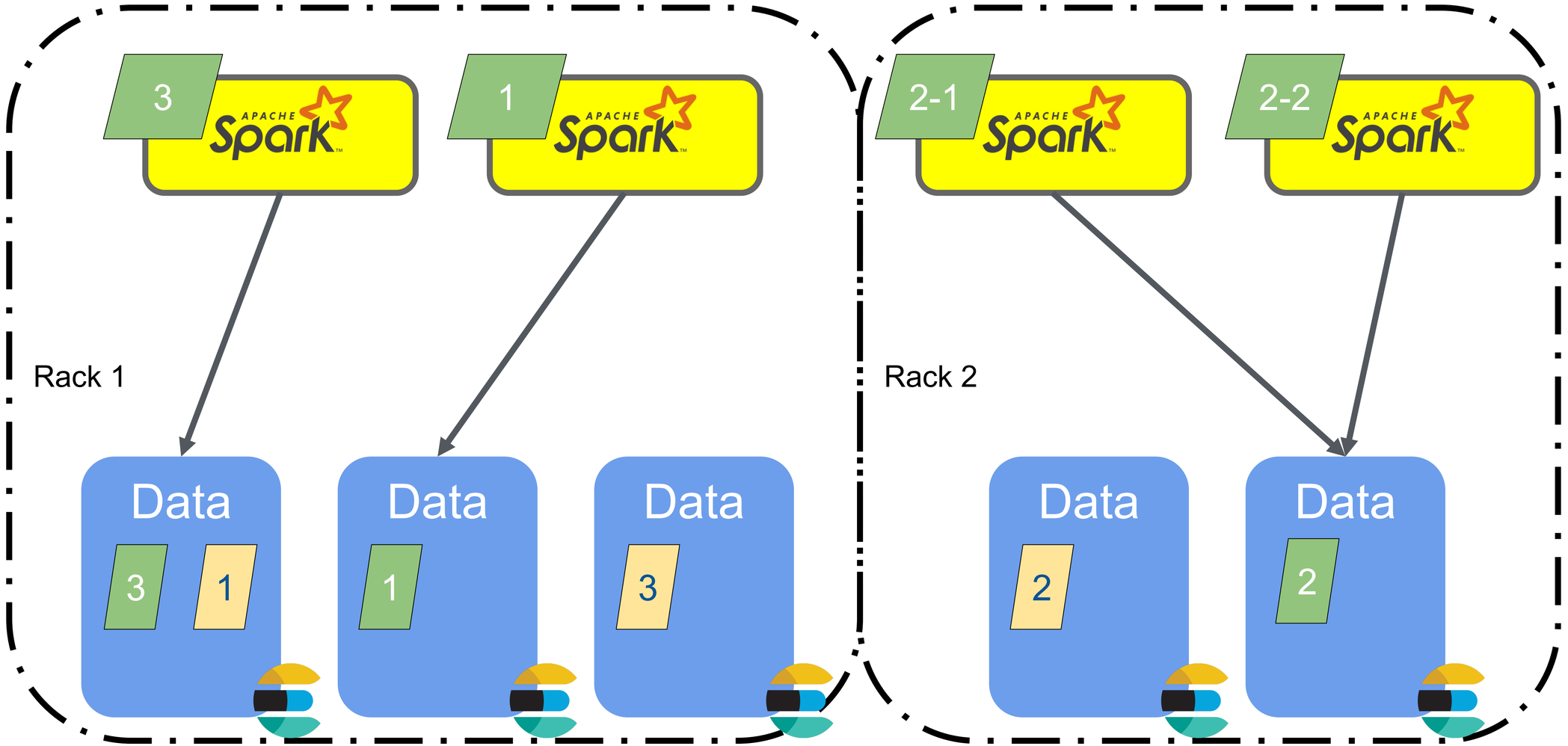
Launching Tasks…



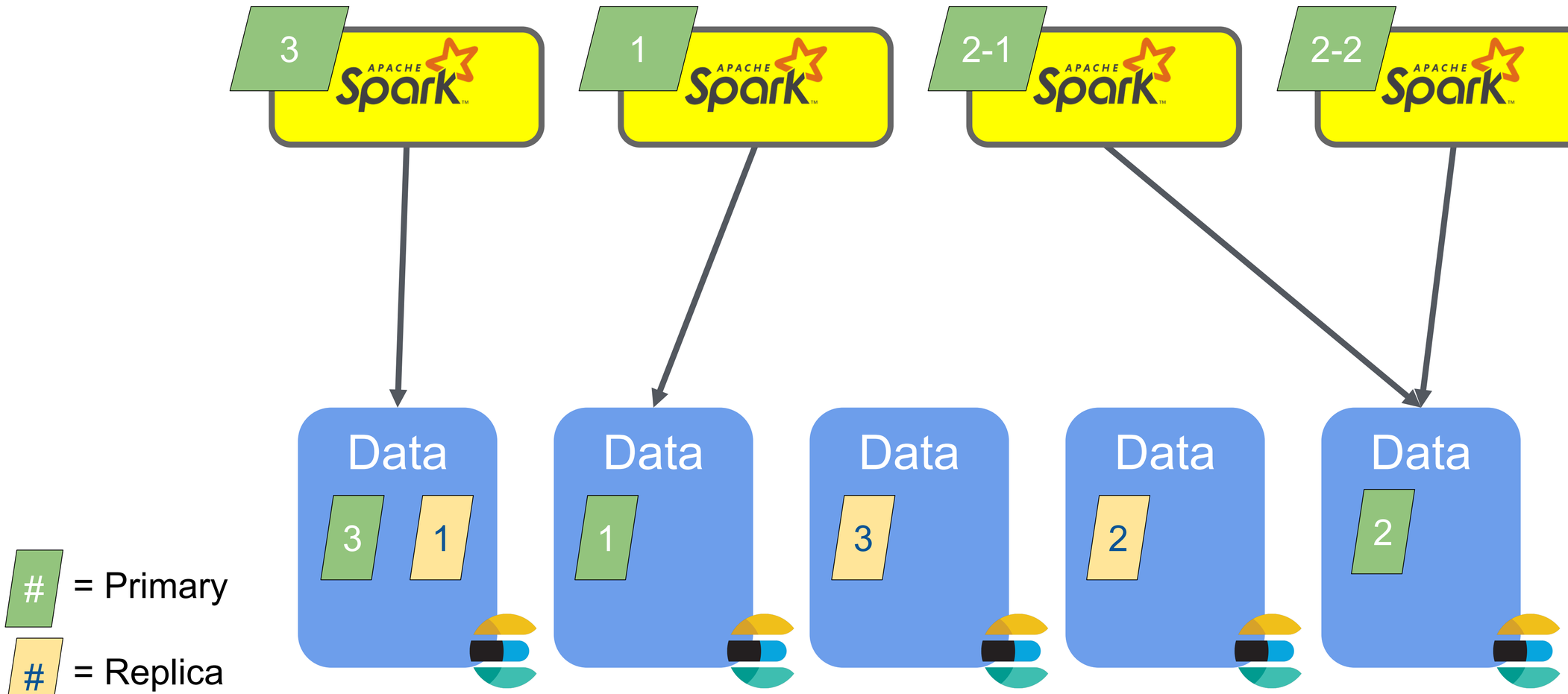
Job Execution



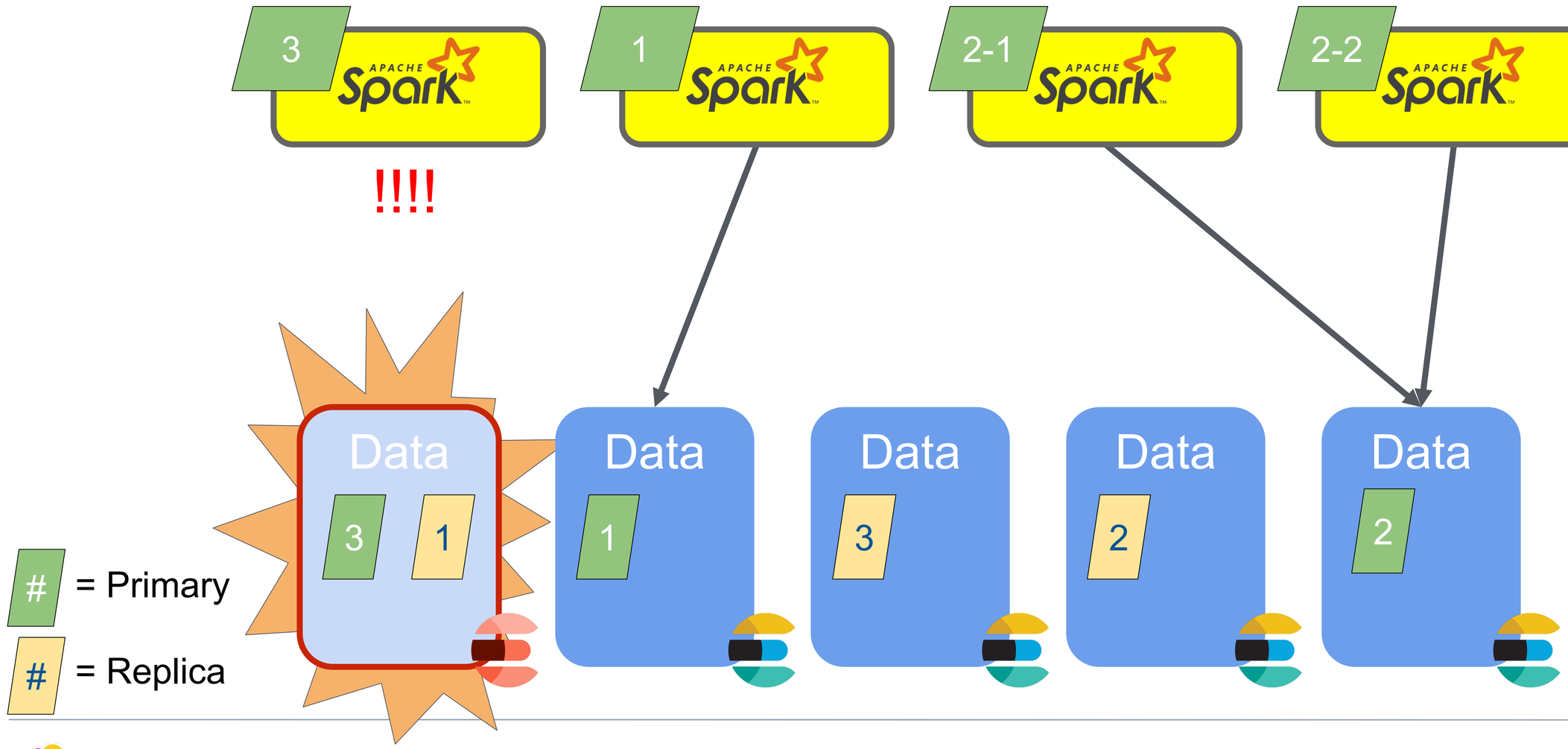
Job Execution



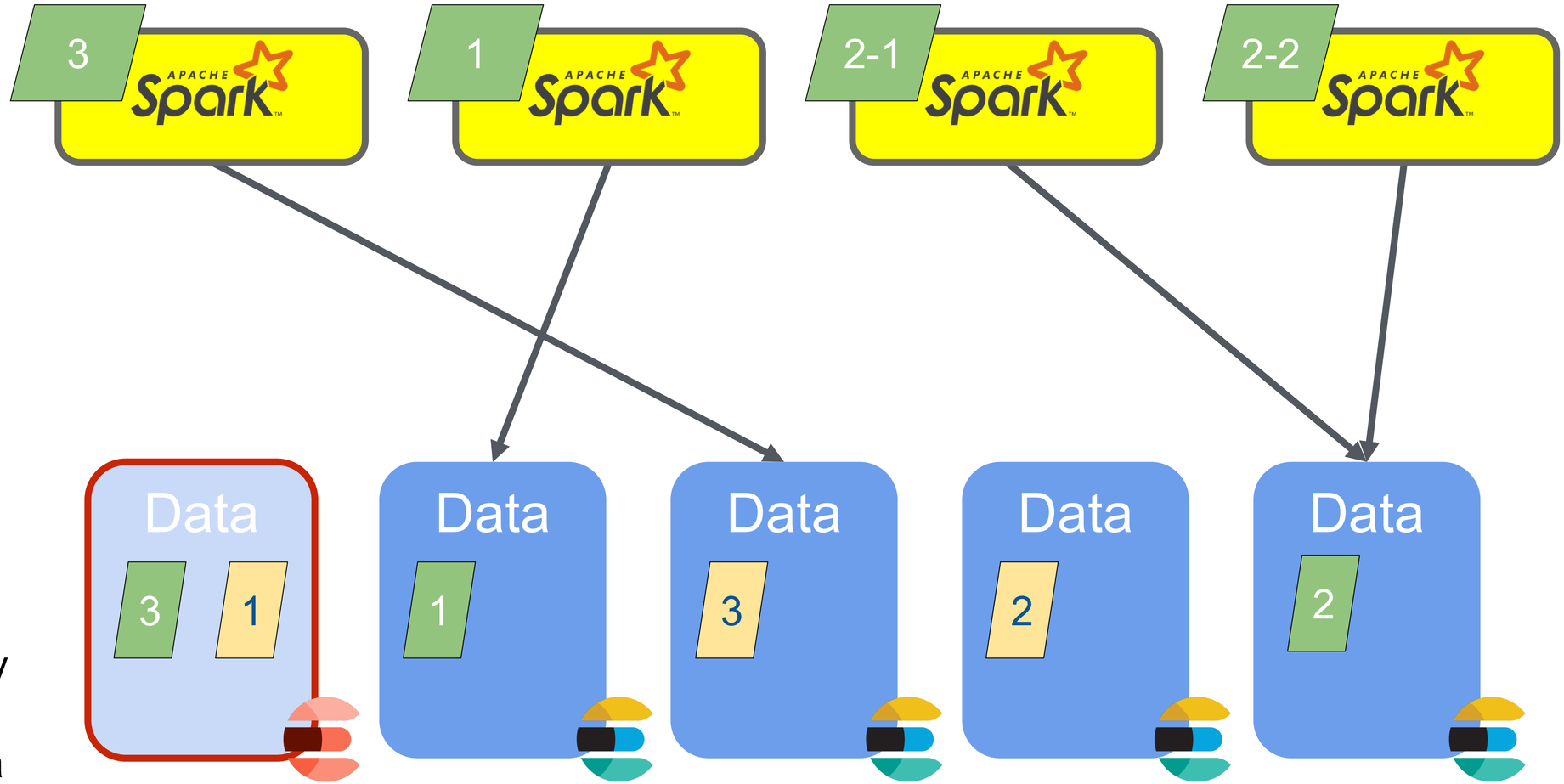
Job Execution



Communication Failure



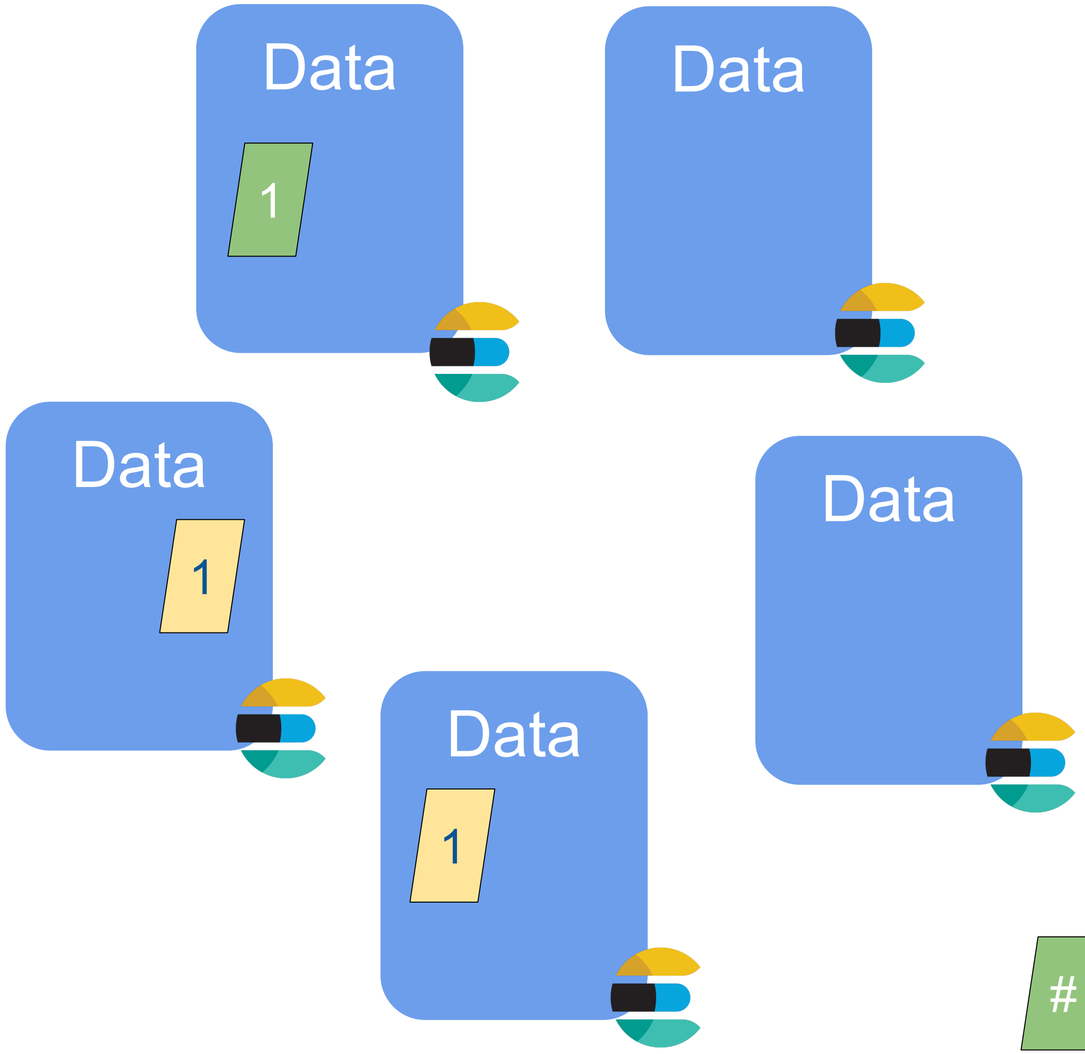
Communication Failure



Writing to Elasticsearch

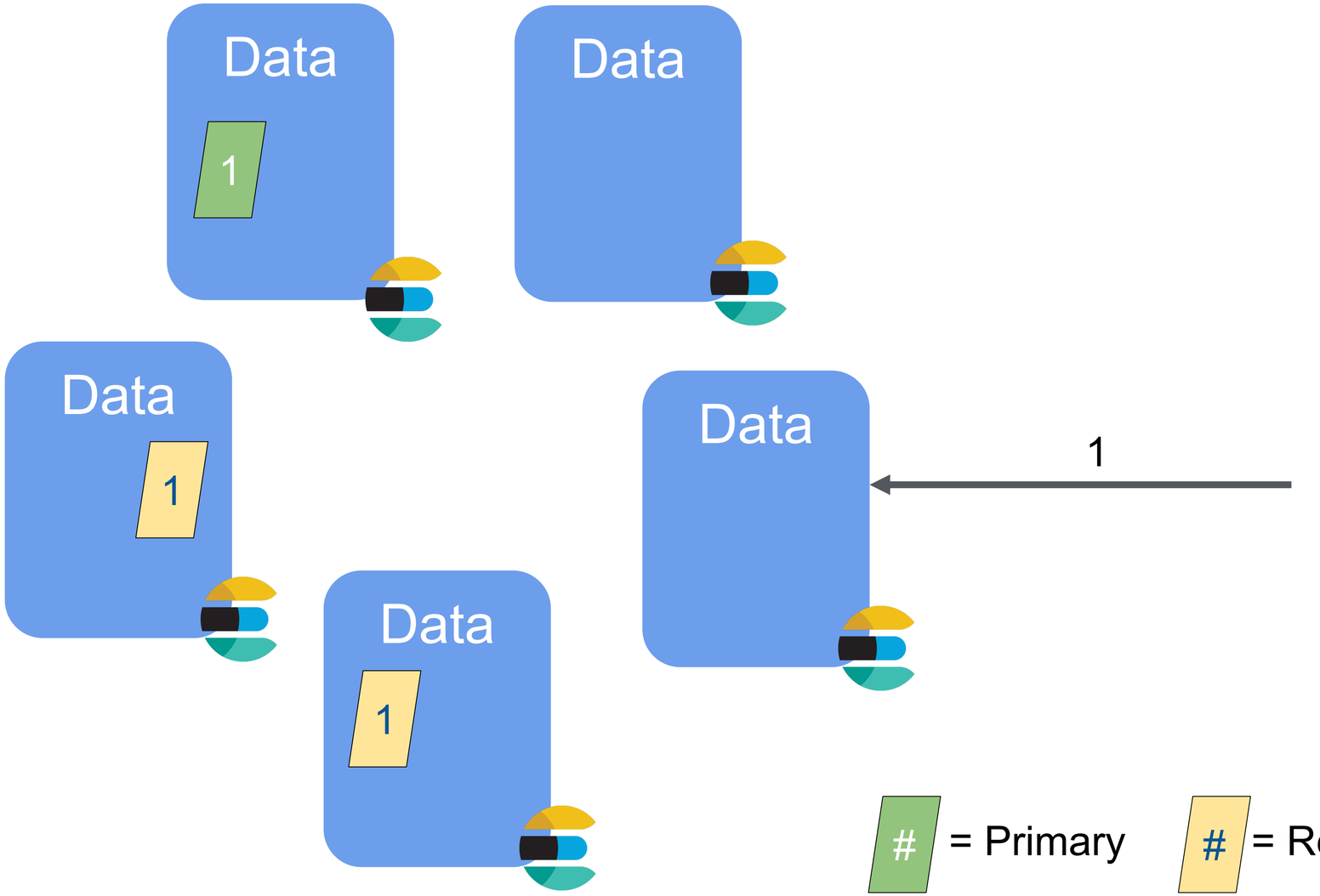
Finding Partitions

Writing to index ...



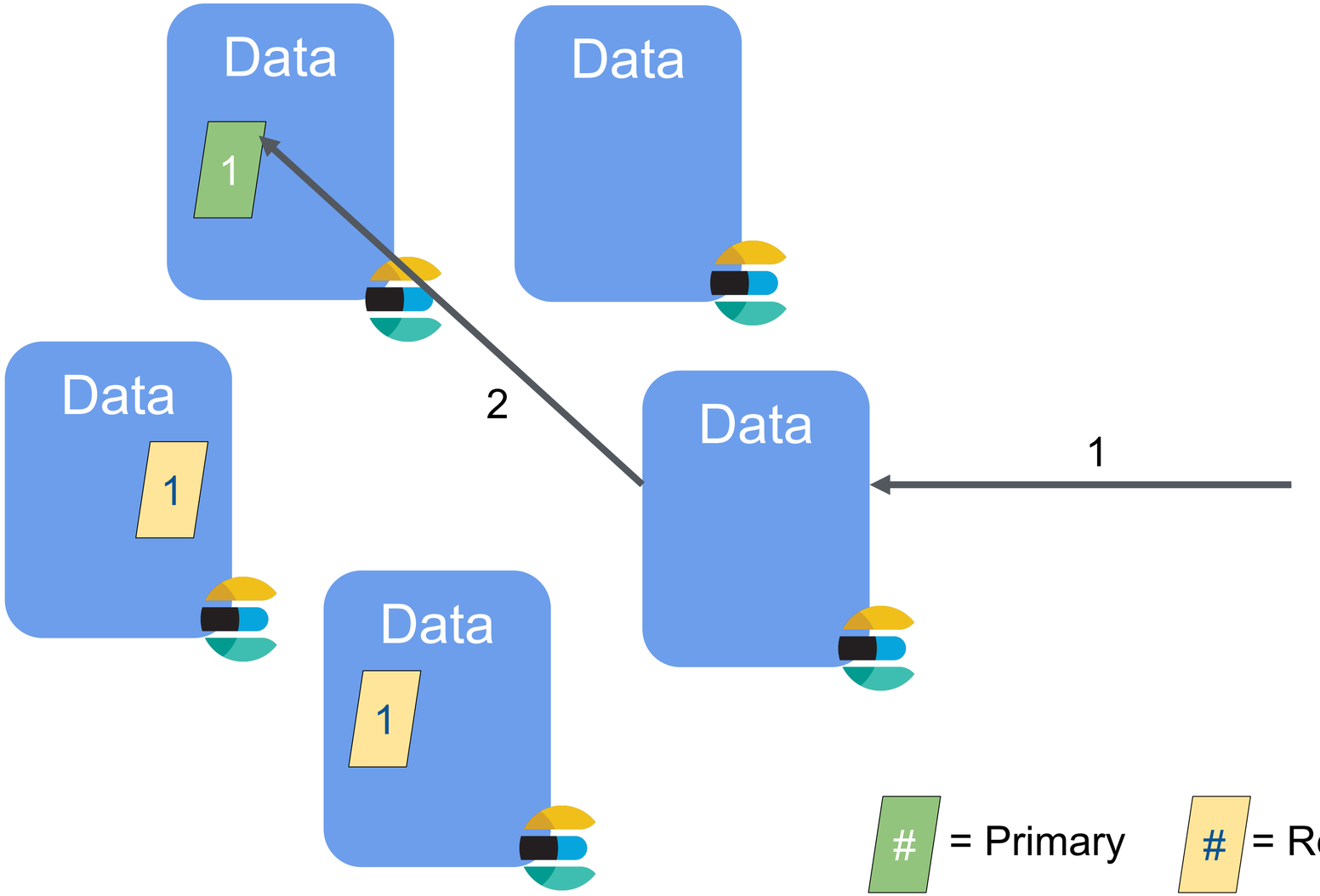
Finding Partitions

Writing to index ...



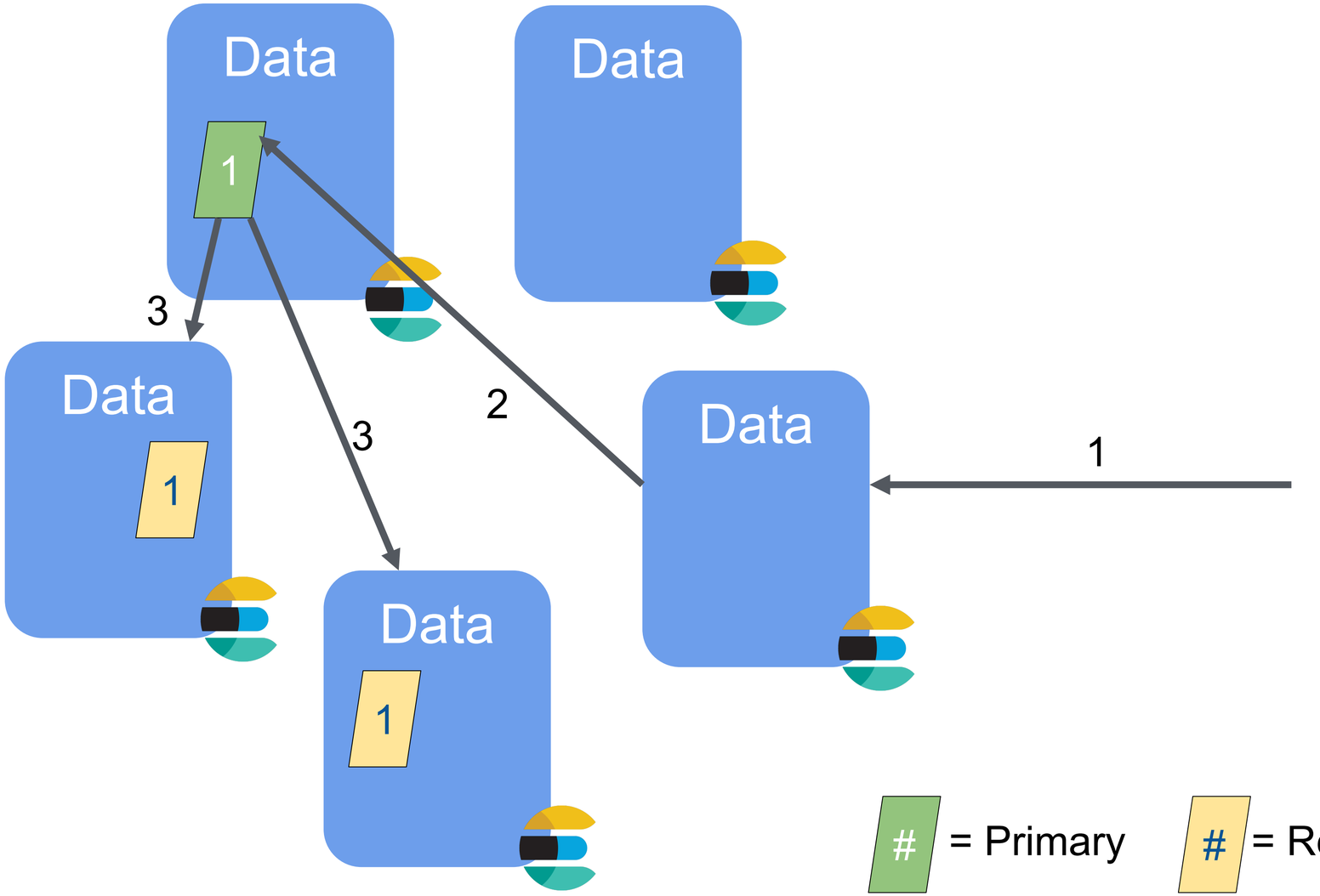
Finding Partitions

Writing to index ...



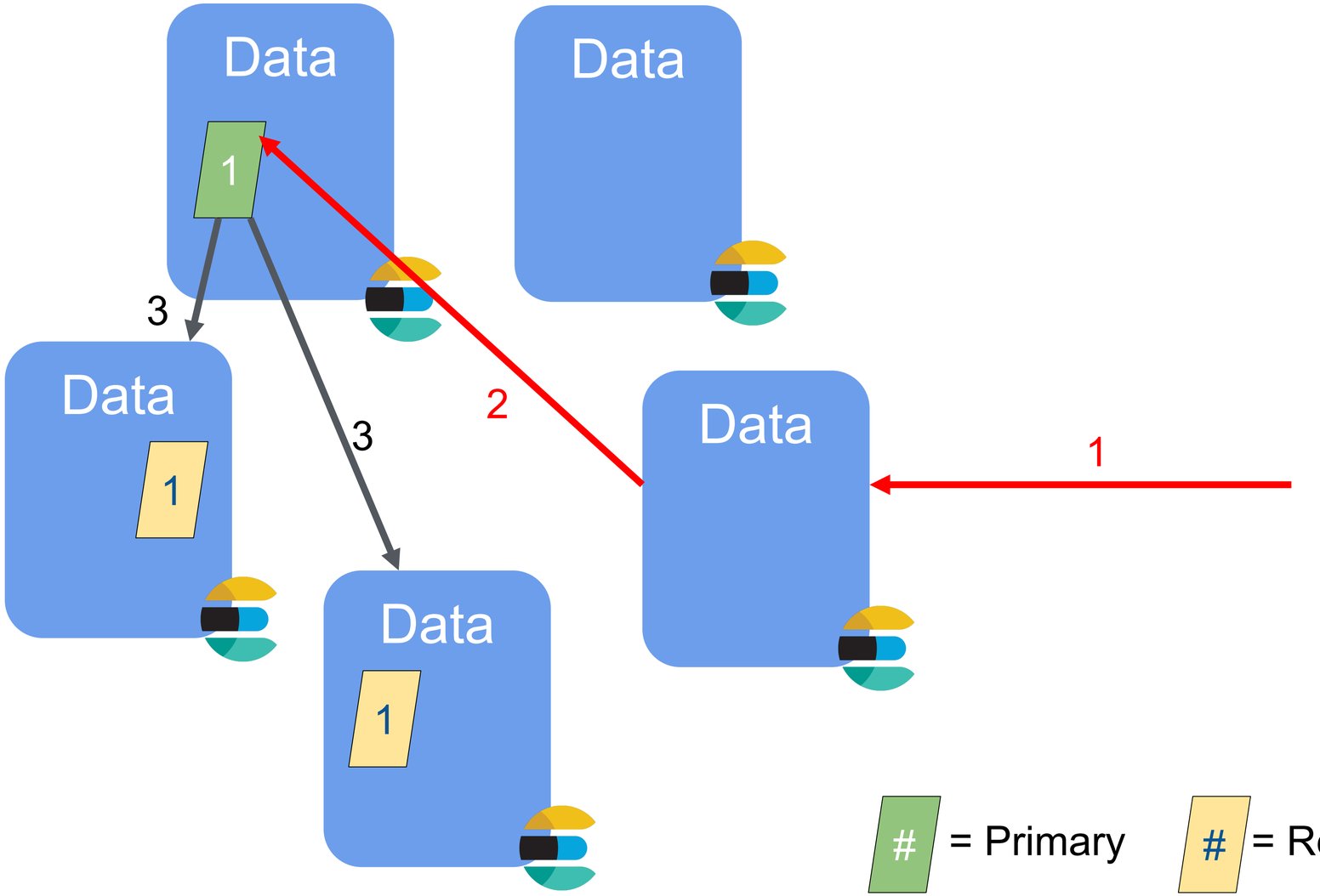
Finding Partitions

Writing to index ...



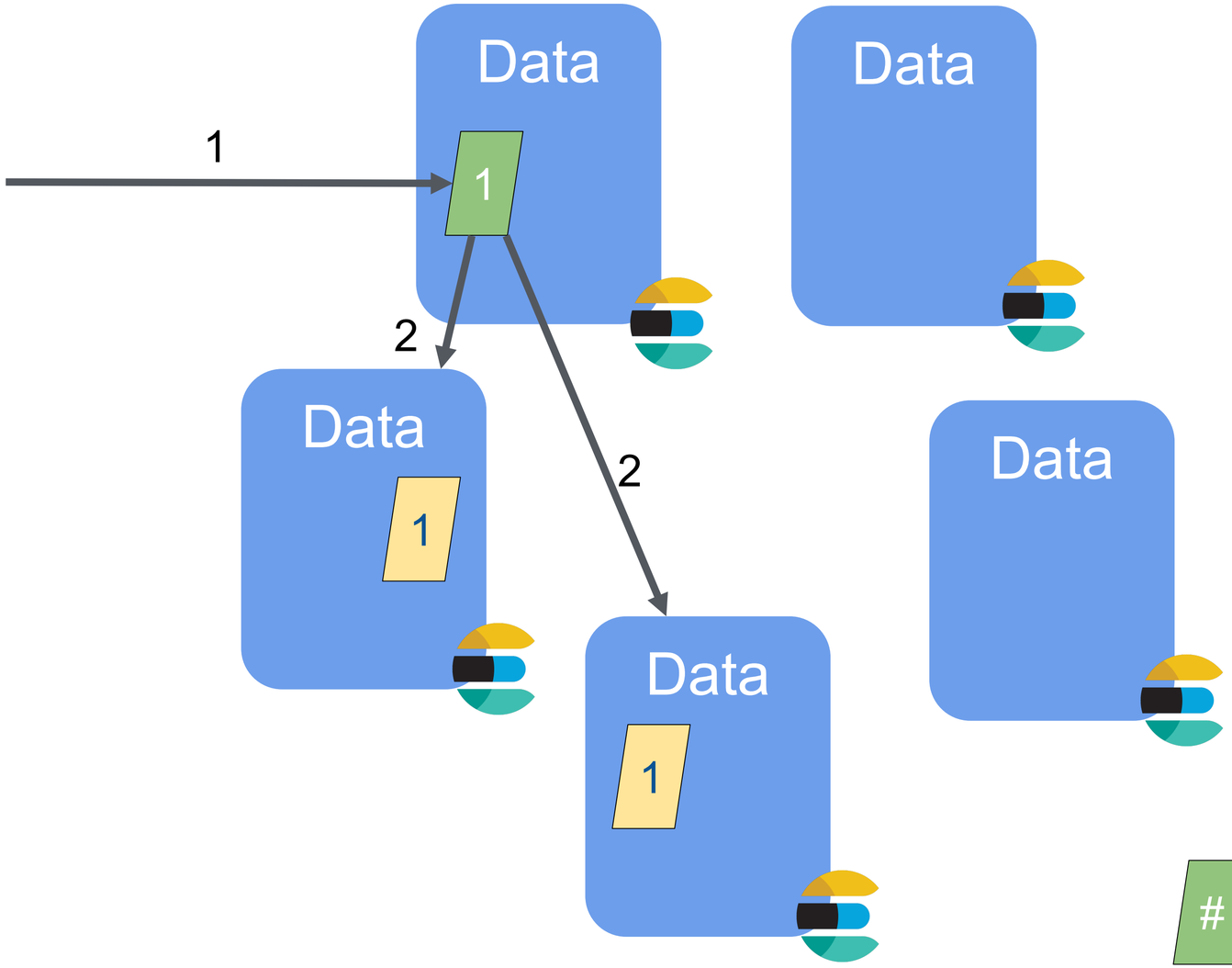
Finding Partitions

Writing to index ...



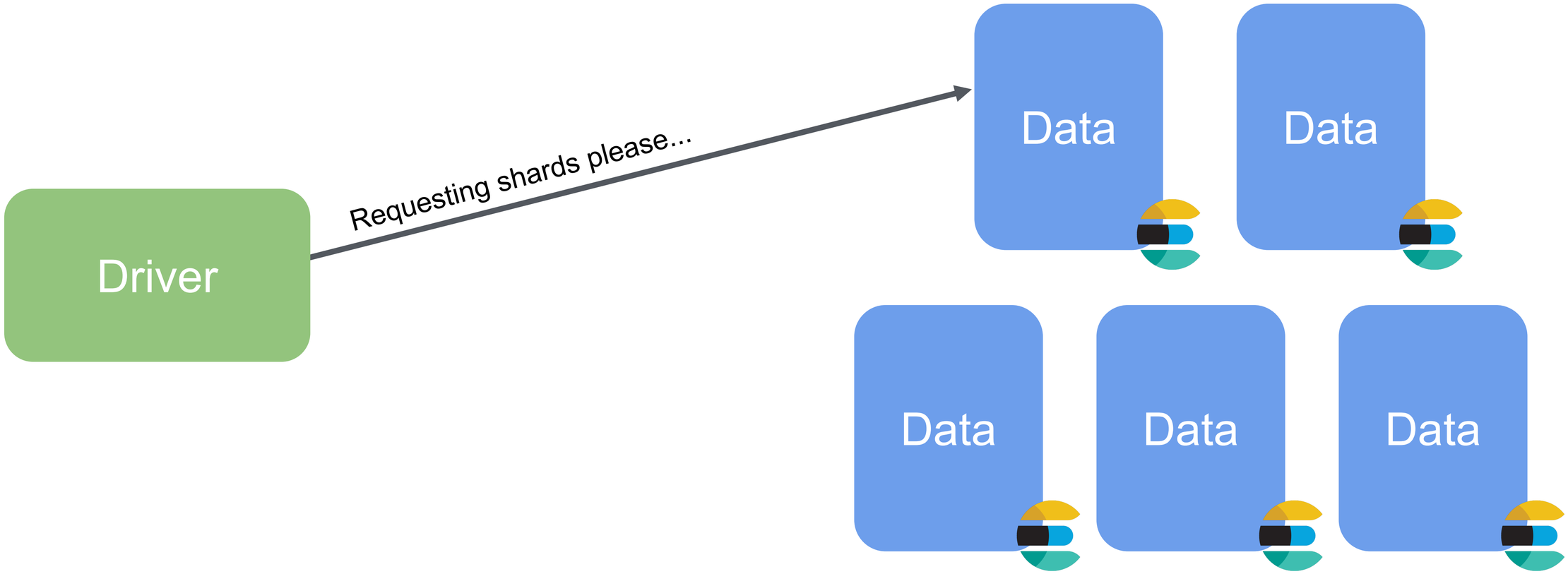
Finding Partitions

Writing to index ...



Finding Partitions

Writing to index ...



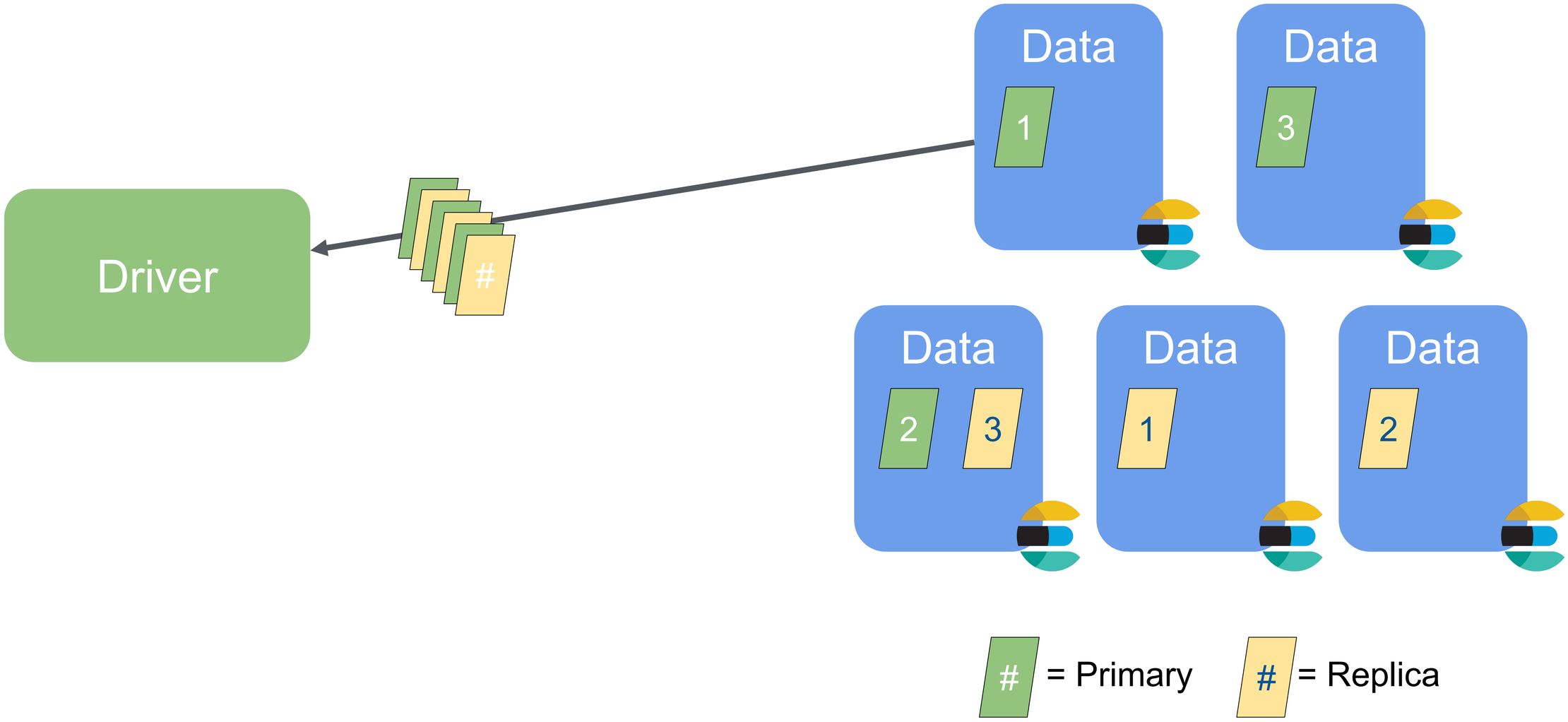
Finding Partitions

Writing to index ...

Driver

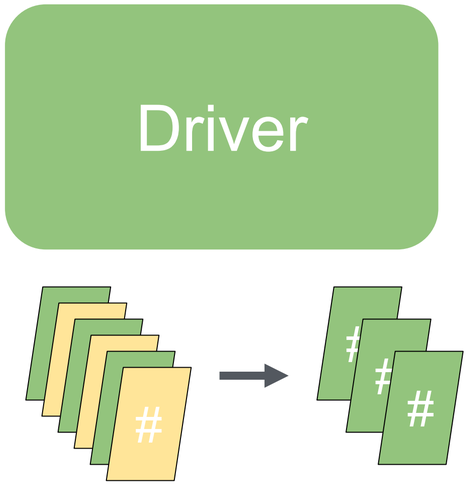
Finding Partitions

Writing to index ...



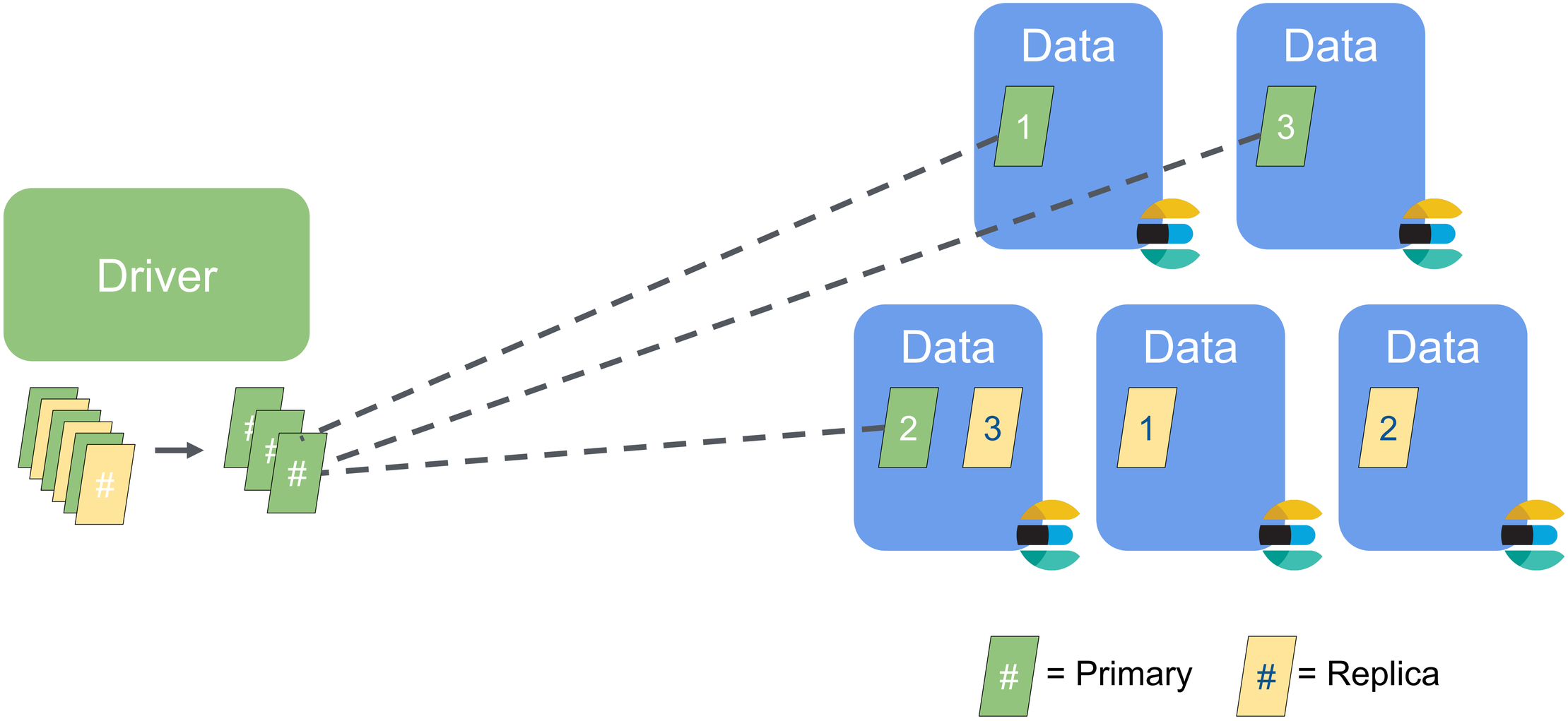
Finding Partitions

Writing to index ...



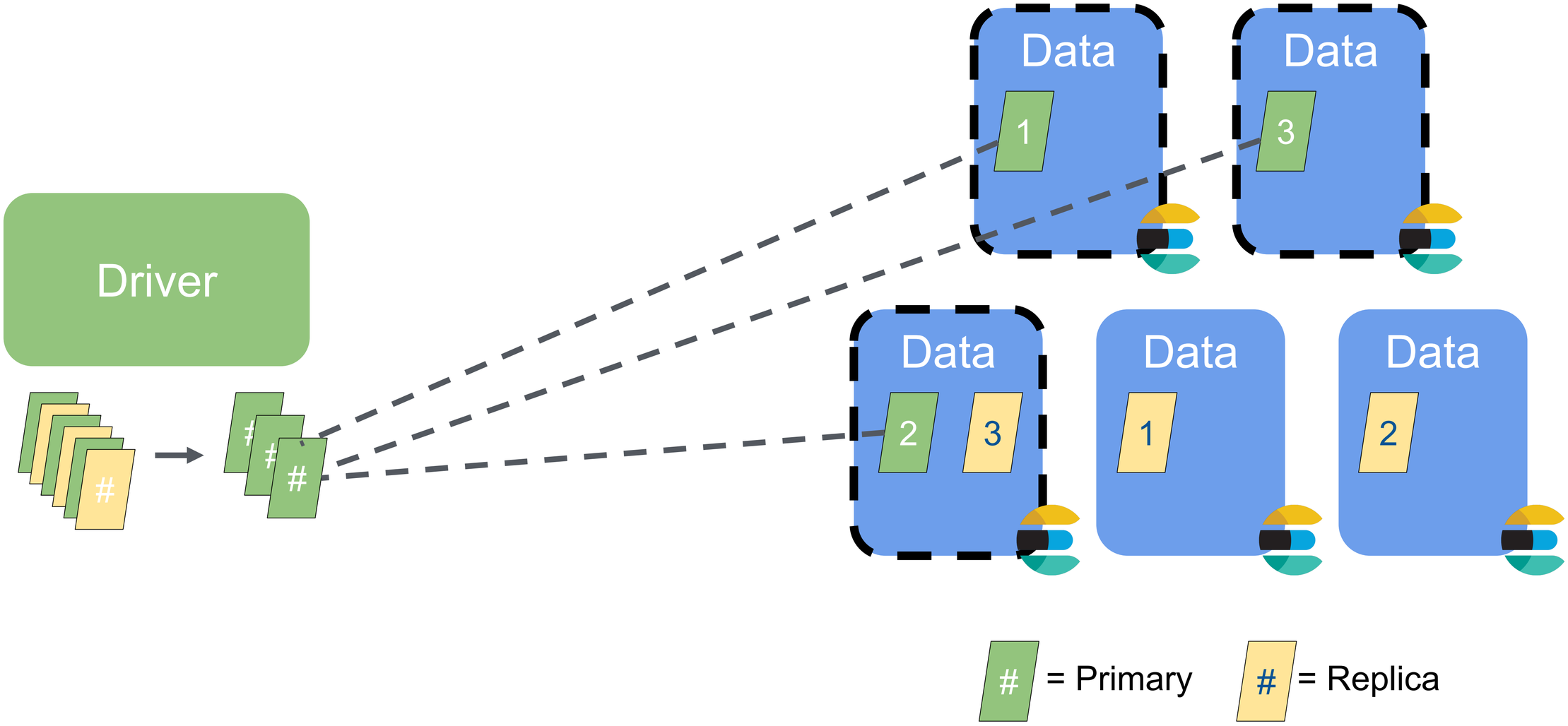
Finding Partitions

Writing to index ...



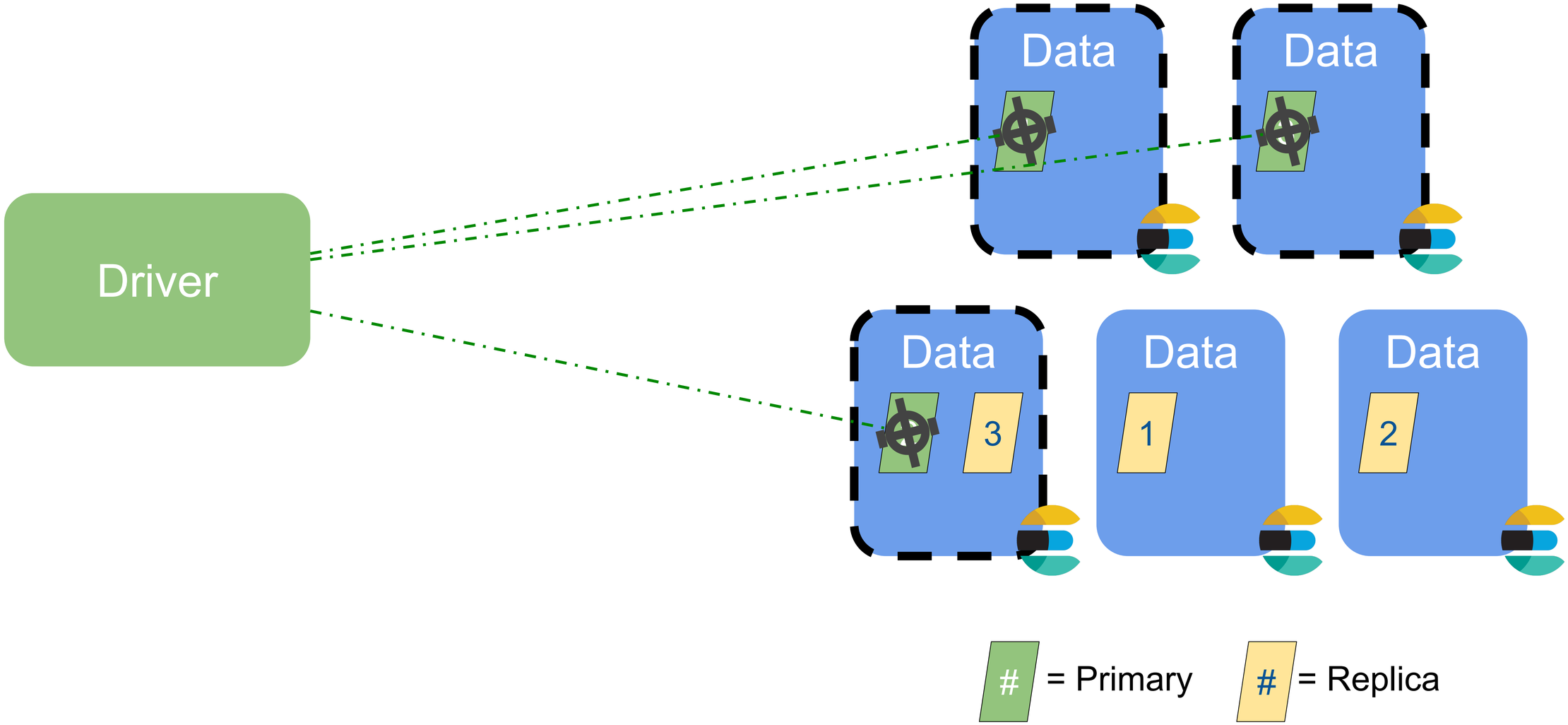
Finding Partitions

Writing to index ...

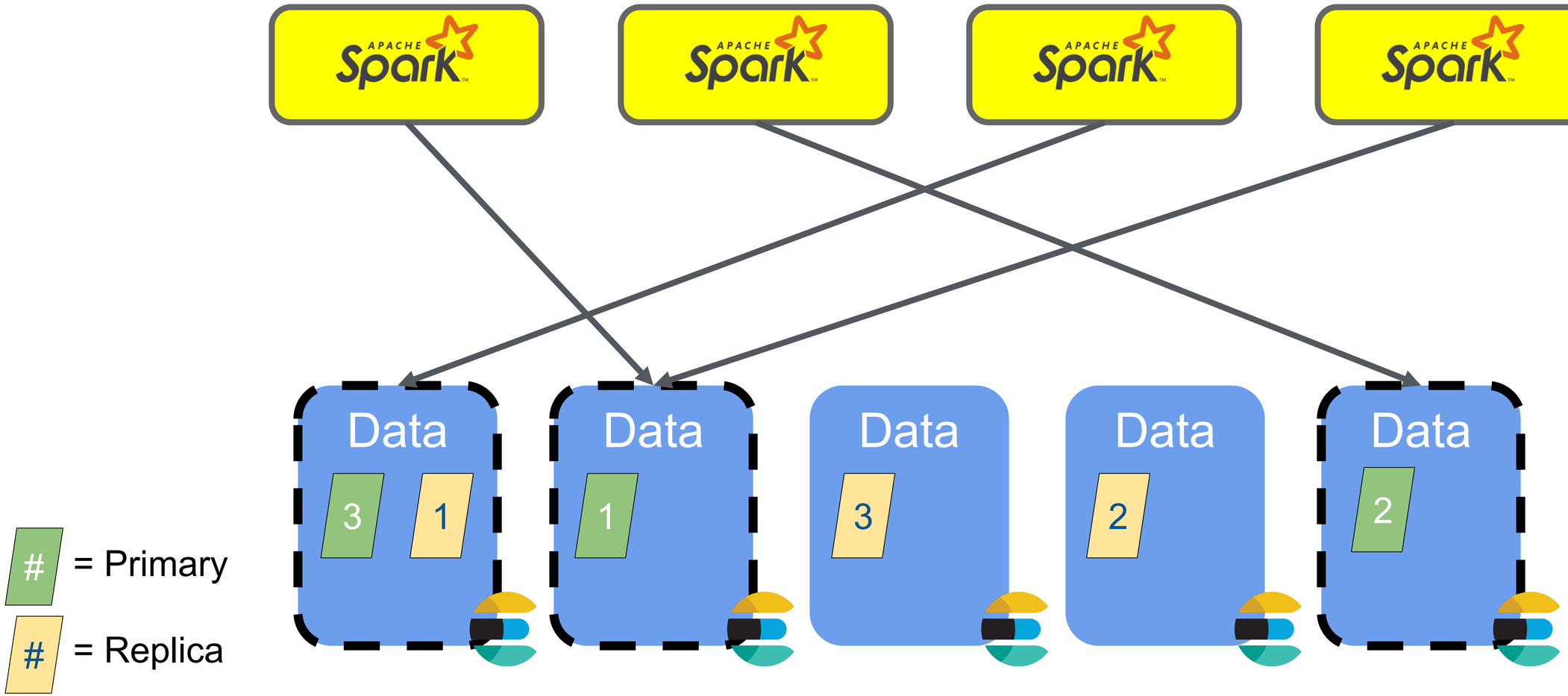


Finding Partitions

Writing to index ...



Job Execution



Feature Tour

Code Examples

Apache Hive

Reading from Elasticsearch

```
CREATE EXTERNAL TABLE artists (
    id       BIGINT,
    name     STRING,
    links    STRUCT<url:STRING, picture:STRING>)
STORED BY 'org.elasticsearch.hadoop.hive.EsStorageHandler'
TBLPROPERTIES(
  'es.resource' = 'radio/artists',
  'es.query' = '?q=me*');

-- stream data from Elasticsearch
SELECT * FROM artists;
```

Apache Pig

Writing to Elasticsearch

```
-- load data from HDFS into Pig using a schema
A = LOAD 'src/test/resources/artists.dat' USING PigStorage()
AS (id:long, name, genre, url:chararray, picture:chararray);

-- transform data
B = FOREACH A GENERATE name, genre, TOTUPLE(url, picture) AS
links;

-- save the result to Elasticsearch
STORE B INTO 'radio/{genre}' USING
org.elasticsearch.hadoop.pig.EsStorage('es.mapping.id=name');
```

Apache Spark

Reading from Elasticsearch

```
import org.elasticsearch.spark._

val conf = ...
val sc = new SparkContext(conf)
val rdd = sc.esRDD("radio/artists")

rdd.take(10)
```

Apache Spark Streaming

DStreams: The Old Way

```
val sc = new SparkContext(conf)
val ssc = new StreamingContext(sc, Seconds(1))

ssc.socketTextStream("127.0.0.1", 9999)
           .foreachRDD(EsSpark.saveToEs(_, "netcat/data"))

ssc.start()
```

Apache Spark Streaming

DStream Native Integration (NEW IN 5.0)

```
import org.elasticsearch.spark.streaming._

val sc = new SparkContext(conf)
val ssc = new StreamingContext(sc, Seconds(1))

ssc.socketTextStream("127.0.0.1", 9999)
           .saveJsonToEs("netcat/data")

ssc.start()
```

Ingest Node Support

Configurations Added for Ingest (NEW IN 5.0)

```
import org.elasticsearch.spark.streaming._

val sc = new SparkContext(conf)
val ssc = new StreamingContext(sc, Seconds(1))

val jobConf = Map("es.ingest.pipeline" -> "hadoop_test")

ssc.socketTextStream("127.0.0.1", 9999)
              .saveJsonToEs("netcat/data", jobConf)

ssc.start()
```

Coming Soon™

- Support for Unicode Index/Type Names
- Source Filtering for RDD's and MapReduce
- User Specified HTTP Headers
- Spark Structured Streaming Integration

Common Use Cases

Anoop Sunke

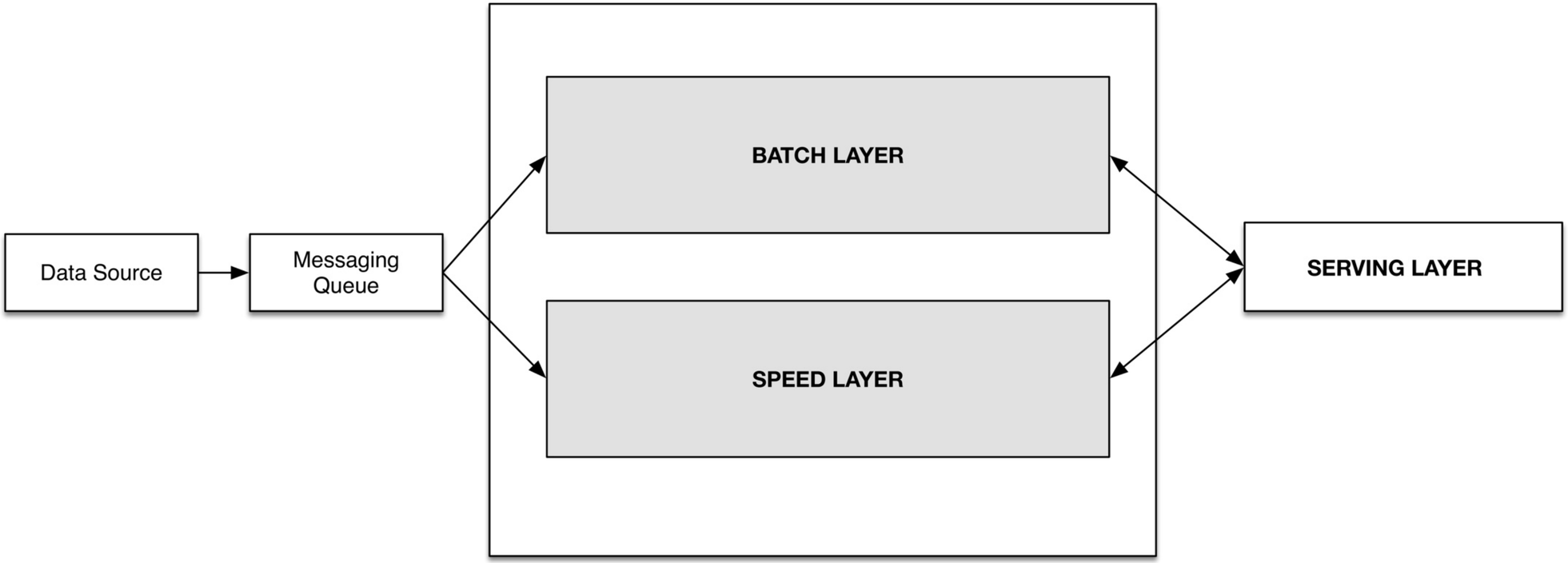
Anoop Sunke

- Solutions Architect @ Elastic
- Ex-Hortonworks, Microsoft
- @apsunke on twitter

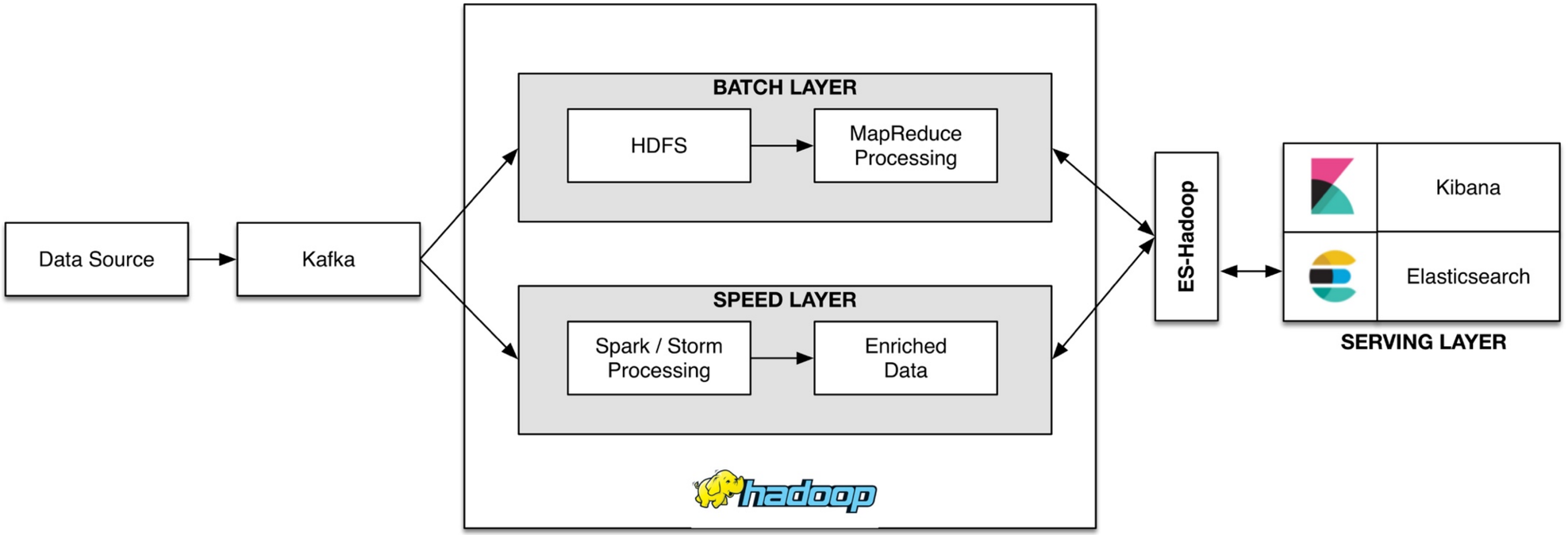
Use Case #1

Lambda Architecture

Lambda Architecture



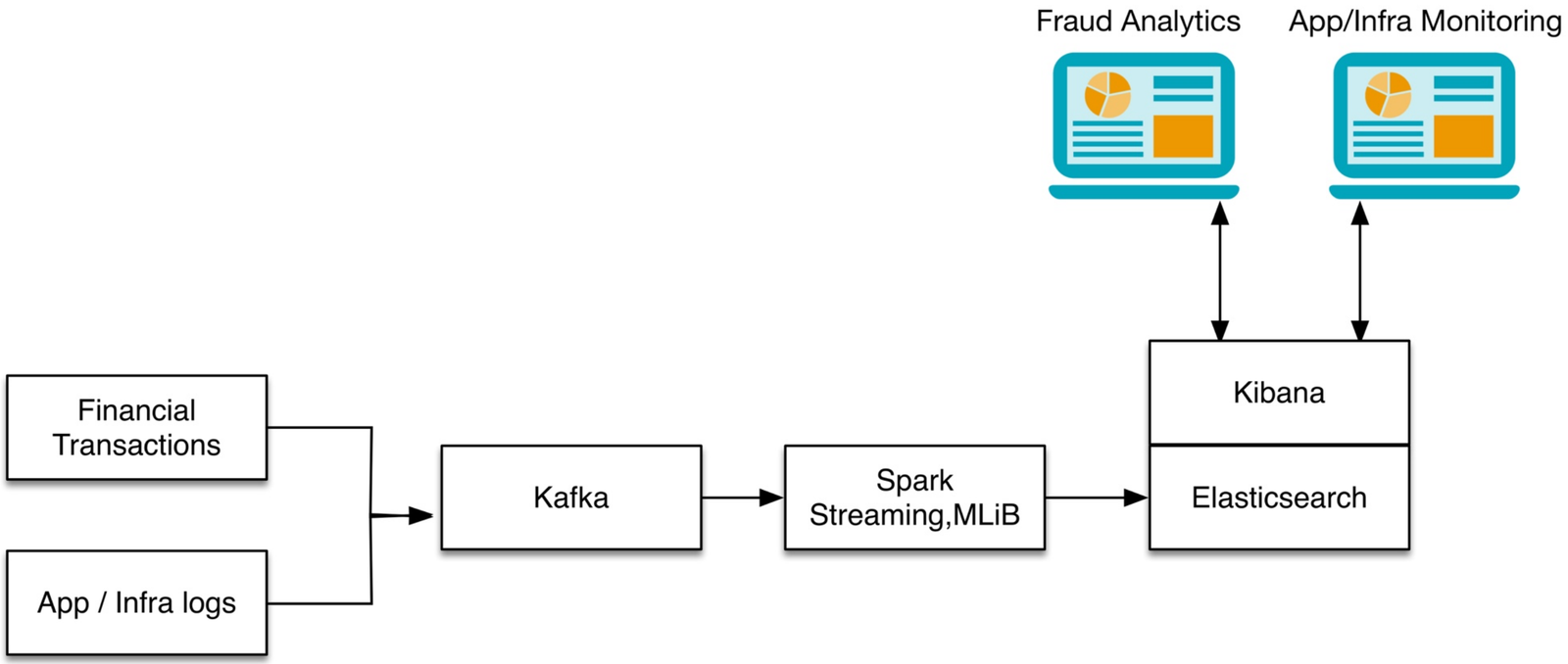
Lambda Architecture



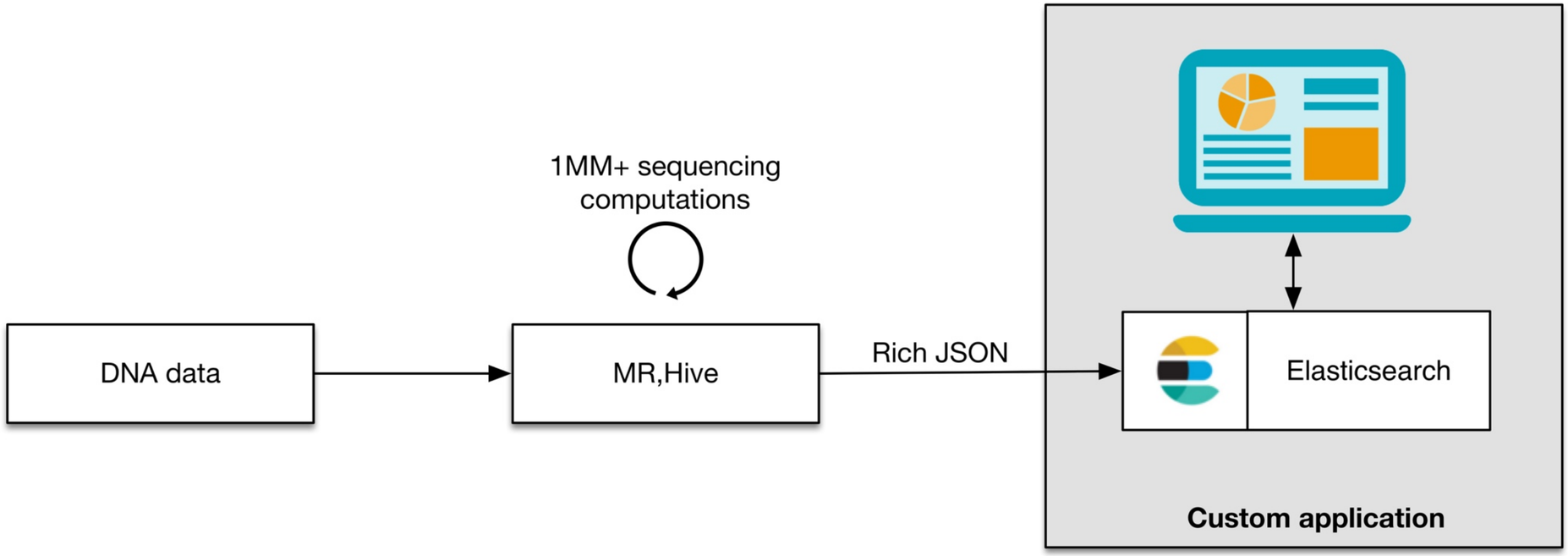
Why ?

- High scale Ingest rate - 1 Million+ Events Per Second
- Data available in 1 second, default refresh interval
- Simple data model – JSON
- Flexible schema - Add fields anytime
- Kibana - Native real time exploration, even at scale
- REST API first data store – Fast Aggregations, Custom UI, Embedding
- Index Index Index !
- Out of the box functionality - sharding, rebalancing, easy scaling

Financial Services company



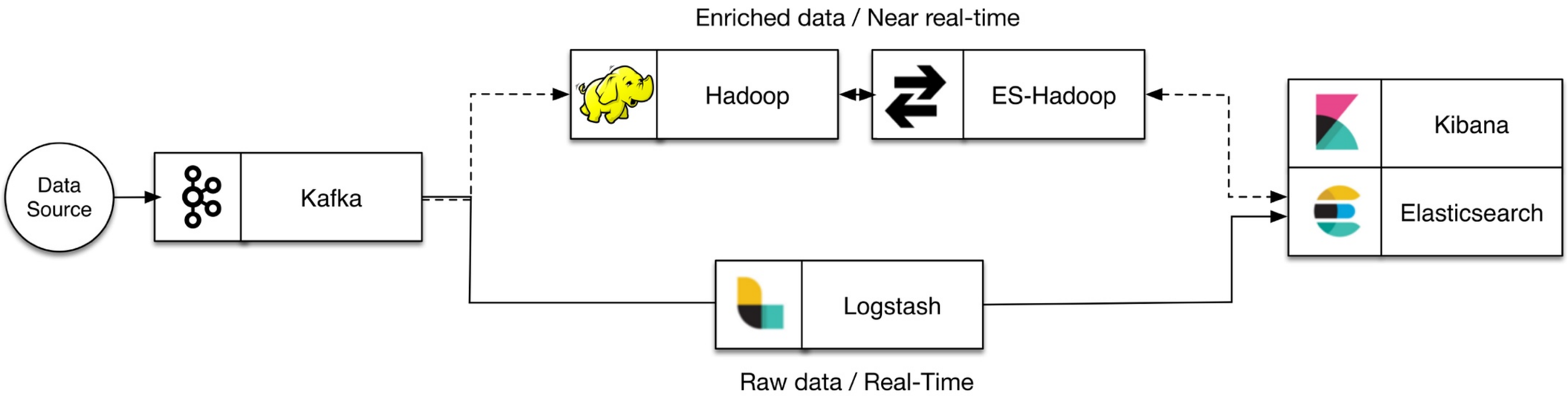
Bioinformatics Company



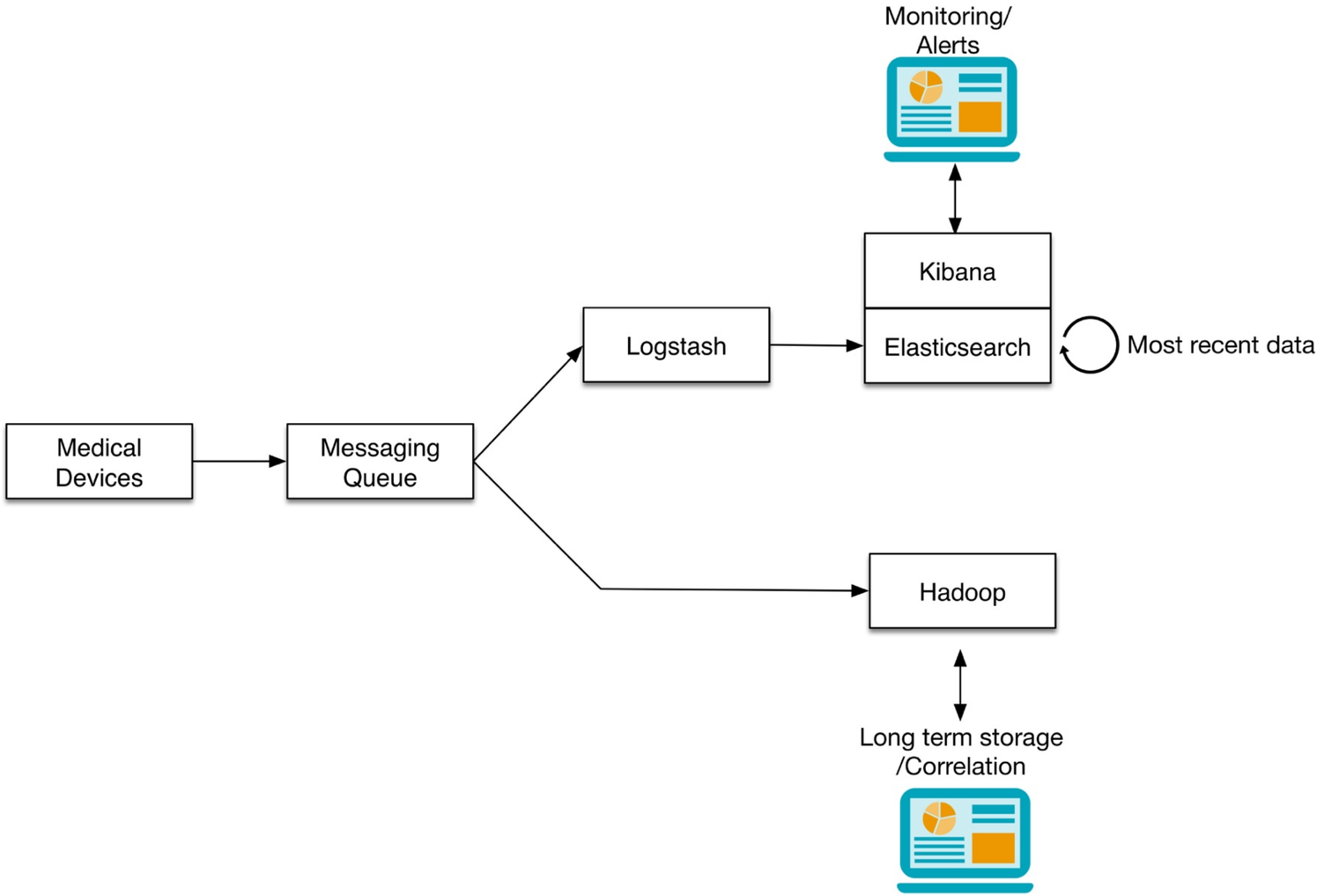
Use Case #2

Raw Real-time

Raw real-time Architecture



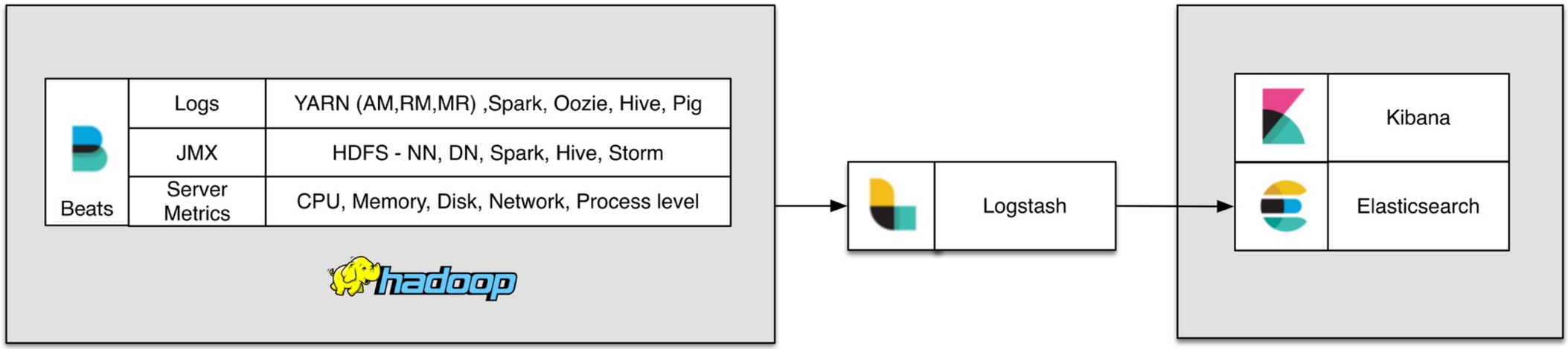
Medical Devices Company



Use Case #3

Monitoring Hadoop

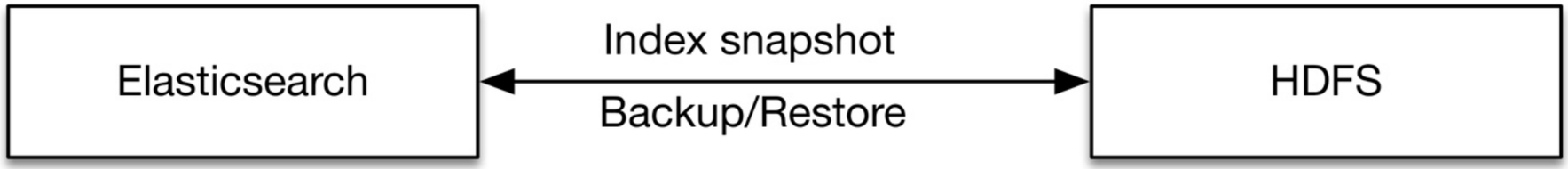
Monitoring Hadoop



Use Case #4

HDFS as a backup store

HDFS as a backup store



Thank You!

Visit us on our forums!

https://discuss.elastic.co/c/elasticsearch-and-hadoop

www.elastic.c
o