دوره جامع تکنولوژی داده (DataTech)

کلان داده

فناپ
FANAP

وزارت علوم، تحقیقات و فناوری
دانشگاه خاتم
Khatam University

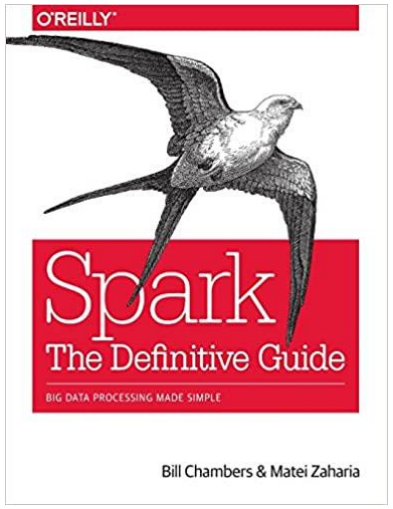
معرفی مدرس

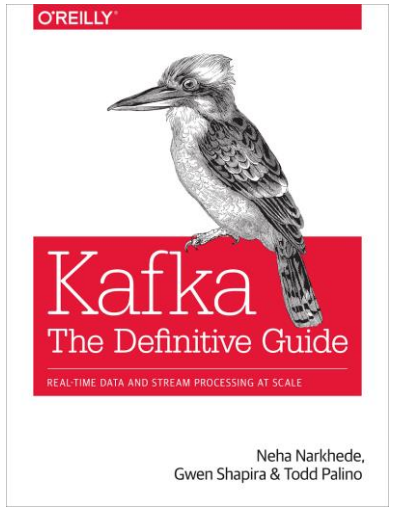
مسلم دلیریان

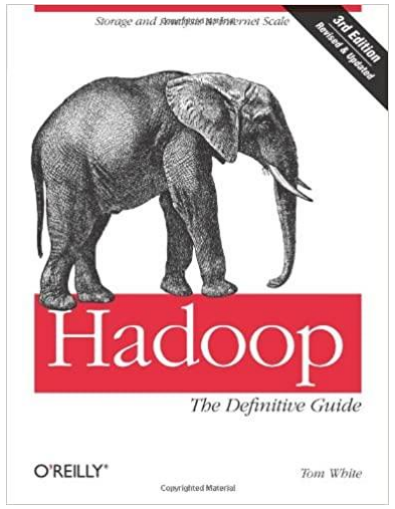
- کارشناس ارشد علوم کامپیوتر
- مدیر تیم مهندسی داده در شرکت فناپ

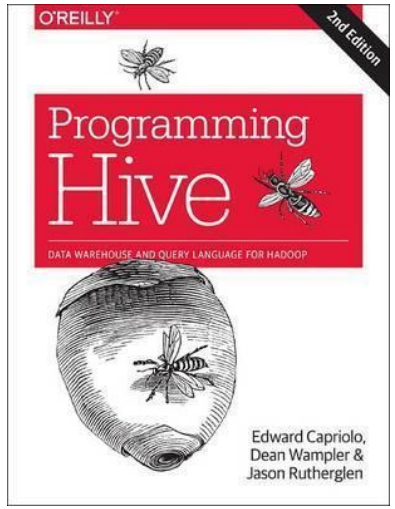
پل های ارتباطی: sm.dalirian@gmail.com

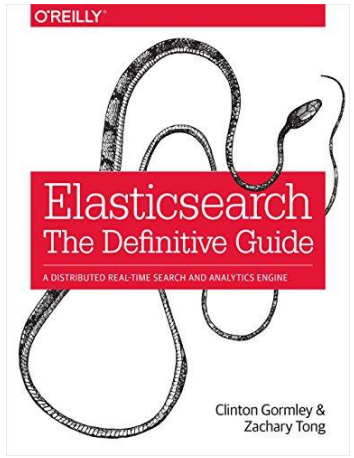
مراجع درس

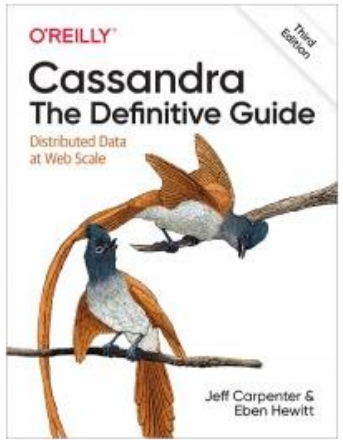

O'REILLY
Spark
The Definitive Guide
BIG DATA PROCESSING MADE SIMPLE
Bill Chambers & Matei Zaharia


O'REILLY
Kafka
The Definitive Guide
REAL-TIME DATA AND STREAM PROCESSING AT SCALE
Neha Narkhede,
Gwen Shapira & Todd Palino


3rd Edition
Hadoop
The Definitive Guide
O'REILLY
Tom White


O'REILLY
2nd Edition
Programming
Hive
DATA WAREHOUSE AND QUERY LANGUAGE FOR HADOOP
Edward Capriolo,
Dean Wampler &
Jason Rutherglen


O'REILLY
Elasticsearch
The Definitive Guide
A DISTRIBUTED REAL-TIME SEARCH AND ANALYTICS ENGINE
Clinton Gormley &
Zachary Tong


O'REILLY
Third Edition
Cassandra
The Definitive Guide
Distributed Data
at Web Scale
Jeff Carpenter &
Eben Hewitt

سرفصل مطالب

- مفاهیم Big Data
- معماری Big Data
- معماری و کار عملی با Hadoop
- معماری و کارعملی با Spark
- معماری و کارعملی با Hive و Impala
- Apache Sentry

سرفصل مطالب

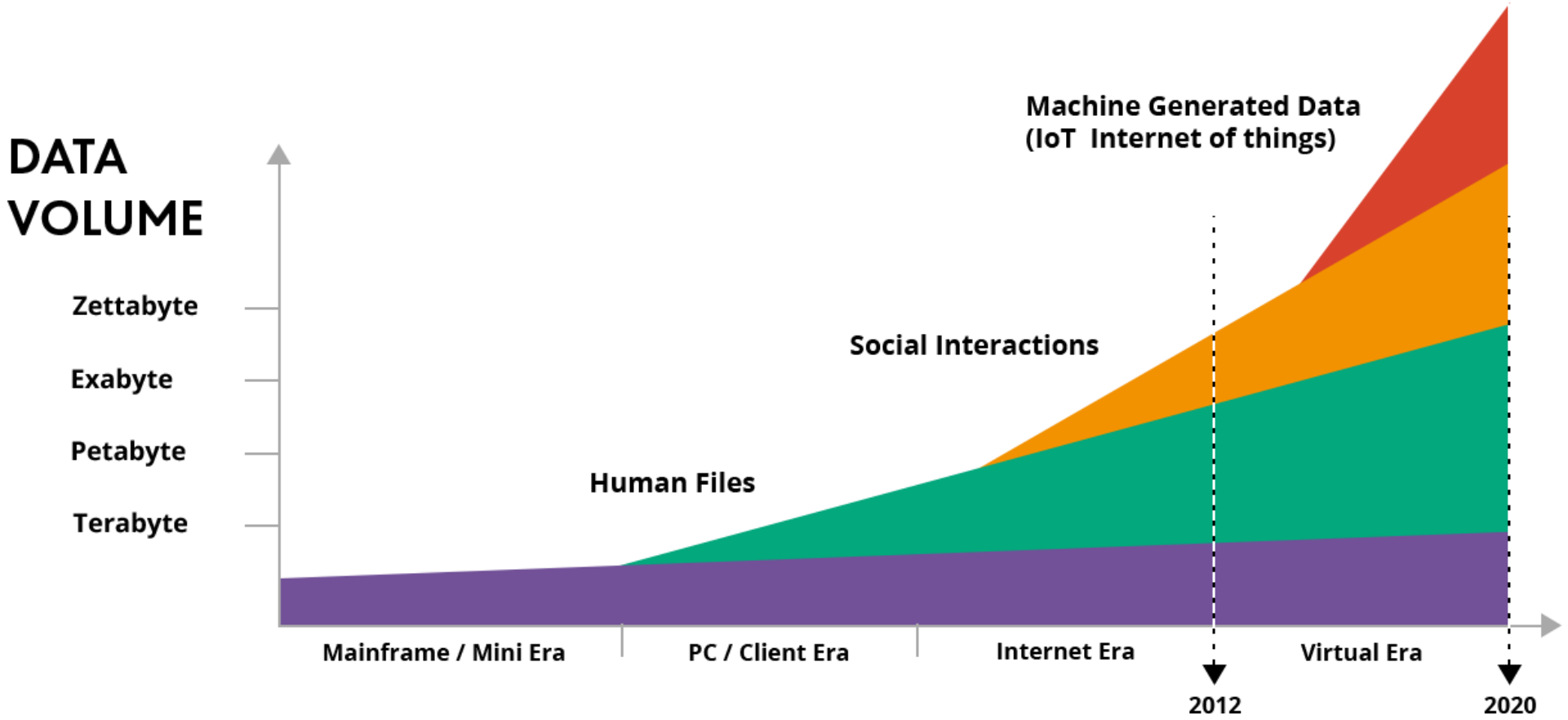
- معماری و کار عملی با Kafka
- معماری NoSQL DB
- جلسه پرسش و پاسخ

مفاهیم Big Data

- داده چیست؟
- داده حجیم چیست؟

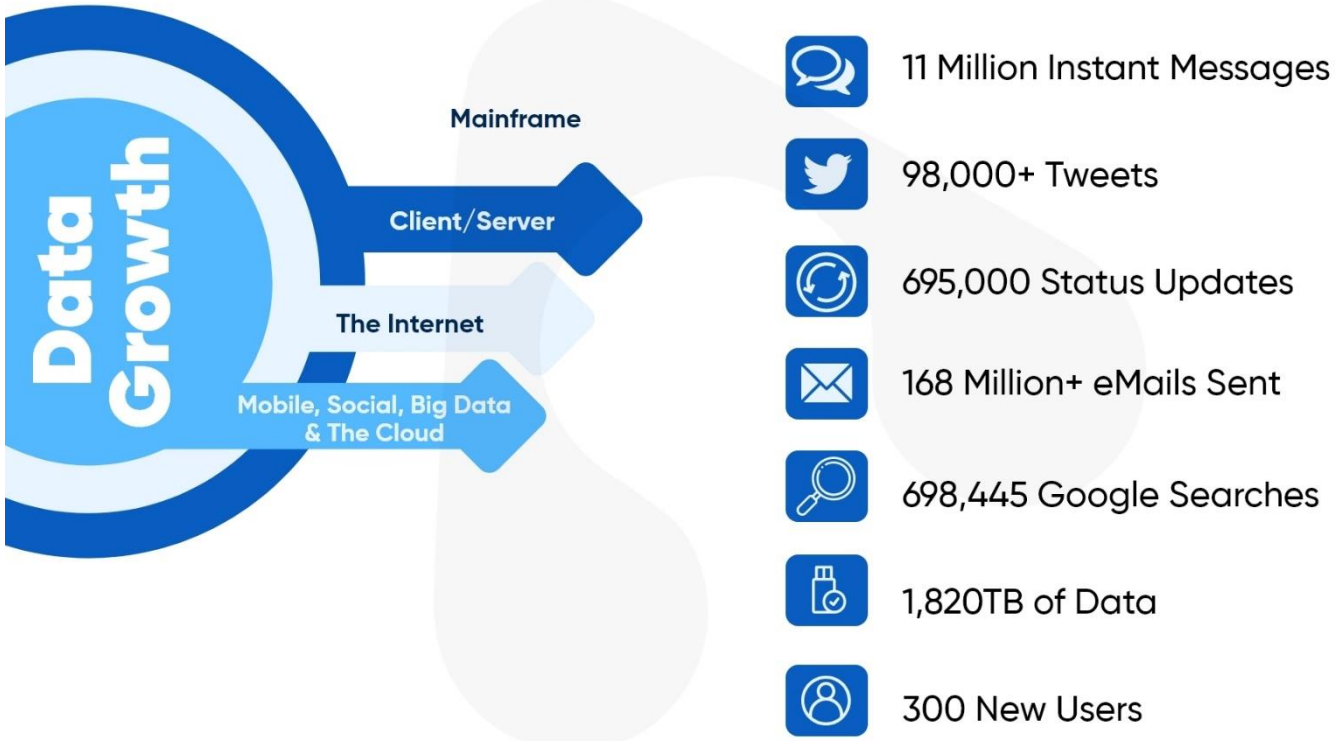
Big Data مفاهیم

- حجم داده‌ها (Volume)



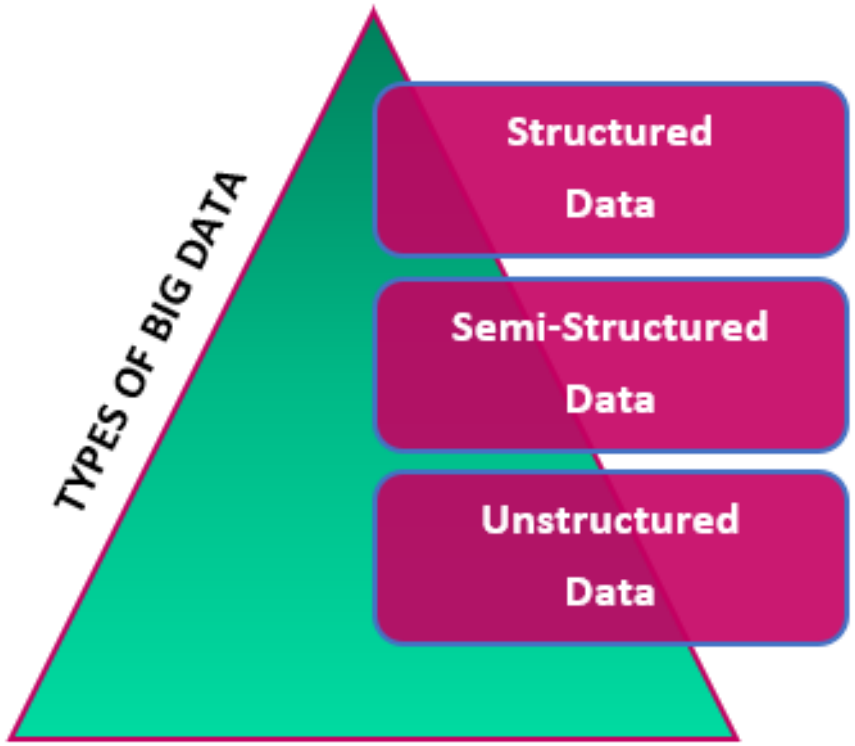
مفاهیم Big Data

- سرعت تولید داده‌ها (Velocity)



Big Data مفاهیم

- انواع داده‌ها (Variety)



Unstructured data

The university has 5600 students.
John's ID is number 1, he is 18 years old and already holds a B.Sc. degree.
David's ID is number 2, he is 31 years old and holds a Ph.D. degree. Robert's ID is number 3, he is 51 years old and also holds the same degree as David, a Ph.D. degree.

Semi-structured data

```
<University>
  <Student ID="1">
    <Name>John</Name>
    <Age>18</Age>
    <Degree>B.Sc.</Degree>
  </Student>
  <Student ID="2">
    <Name>David</Name>
    <Age>31</Age>
    <Degree>Ph.D. </Degree>
  </Student>
  ....
</University>
```

Structured data

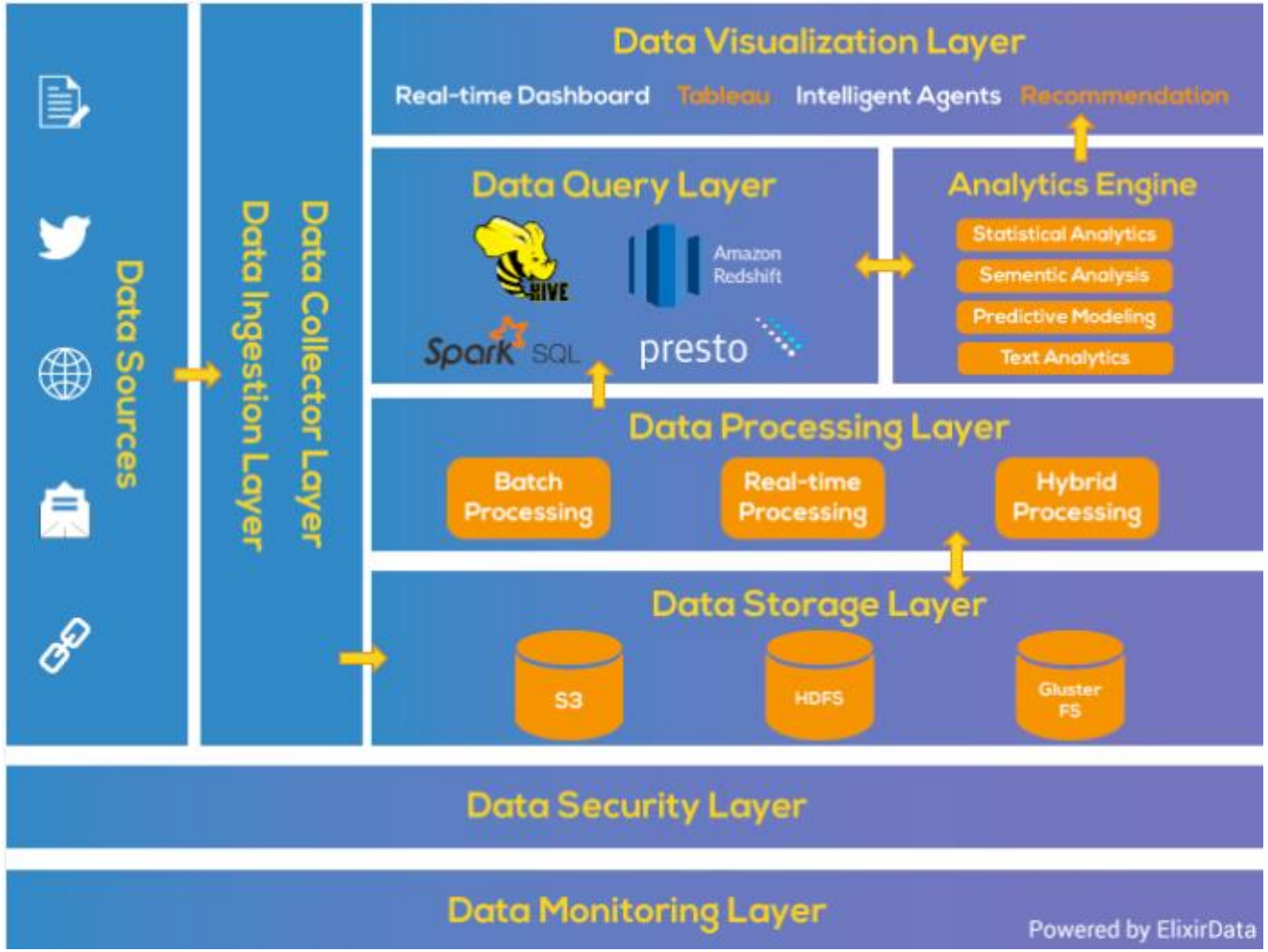
ID	Name	Age	Degree
1	John	18	B.Sc.
2	David	31	Ph.D.
3	Robert	51	Ph.D.
4	Rick	26	M.Sc.
5	Michael	19	B.Sc.

مفاهیم Big Data

- داده‌ها در منابع مختلفی هستند. چگونه باید جمع‌آوری شوند؟
- داده‌ها انواع مختلفی دارند. ذخیره‌سازی داده‌ها چگونه باید انجام شود؟
- صحت و امنیت داده‌ها بسیار مهم است. این مهم چگونه باید تامین شود؟
- برای پردازش داده‌ها نیاز به منابع می‌باشد. بستر مناسب برای پردازش چیست؟
- داده‌ها باید آنالیز شوند. ابزارهای مناسب آنالیز داده در چه بستری قرار دارند؟

معماری Big Data

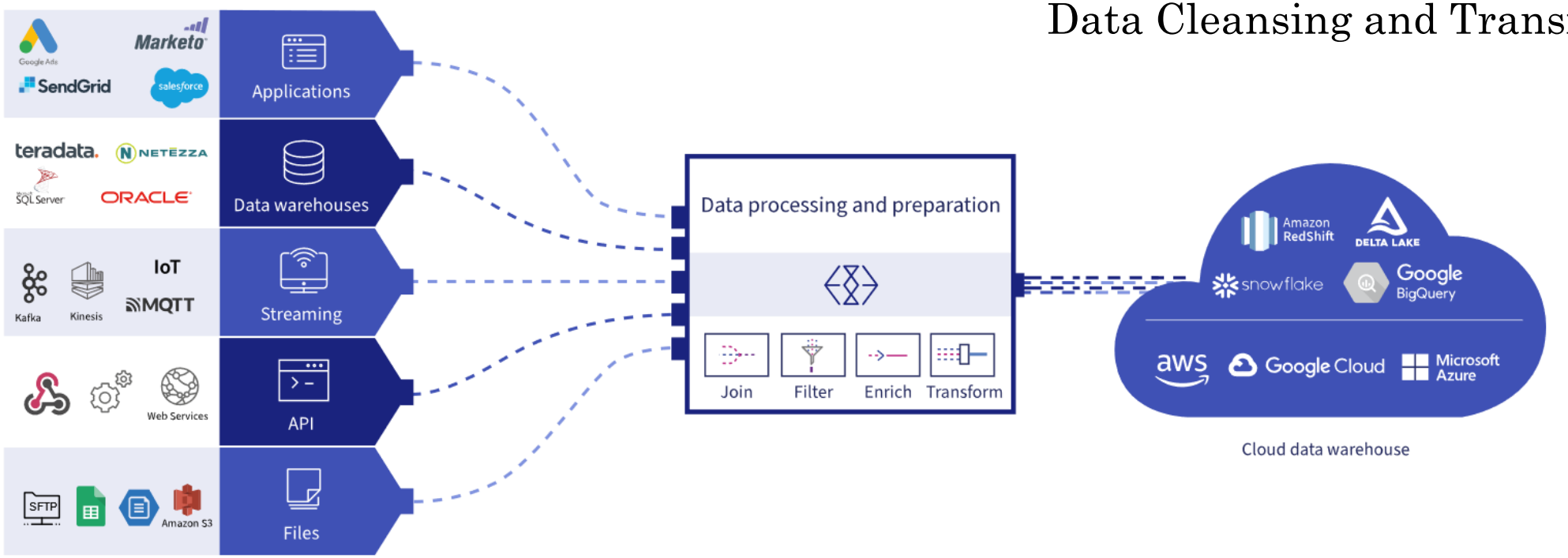
- معماری چند لایه
  Big Data



Big Data معماری

لایه ورود داده‌ها

- Multiple Sources
- Data Cleansing and Transform



معماری Big Data

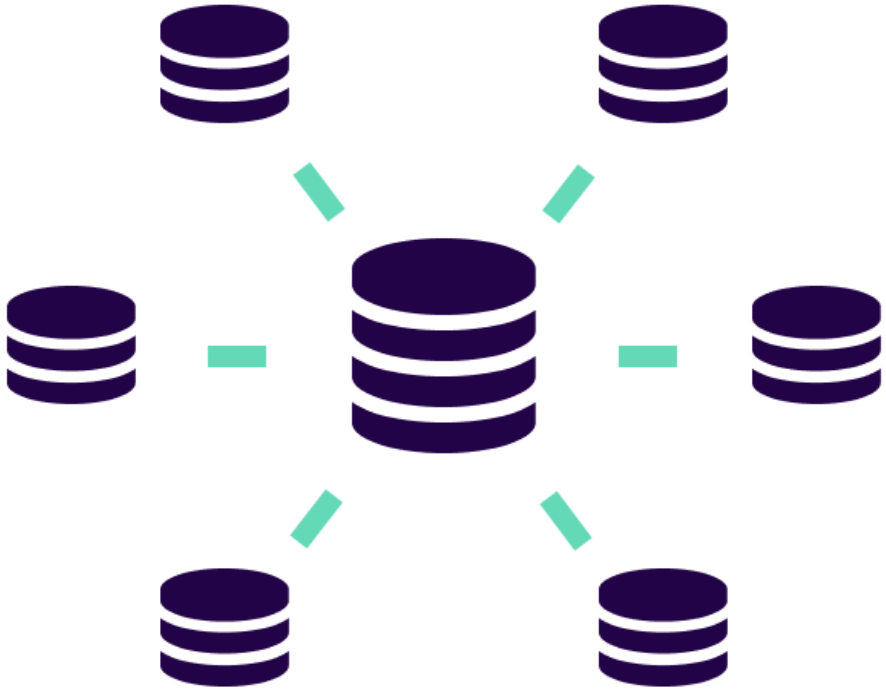
لایه ذخیره‌سازی داده‌ها

- Data Replication
- Data Availability
- Scale out

معماری Big Data

لایه پردازش داده‌ها

- Distributed Processing
- Resource Management



لایه پرس‌وجو داده‌ها

- Select data
- Analytical Query

معماری Big Data

لایه امنیت داده

لایه نمایش نتایج

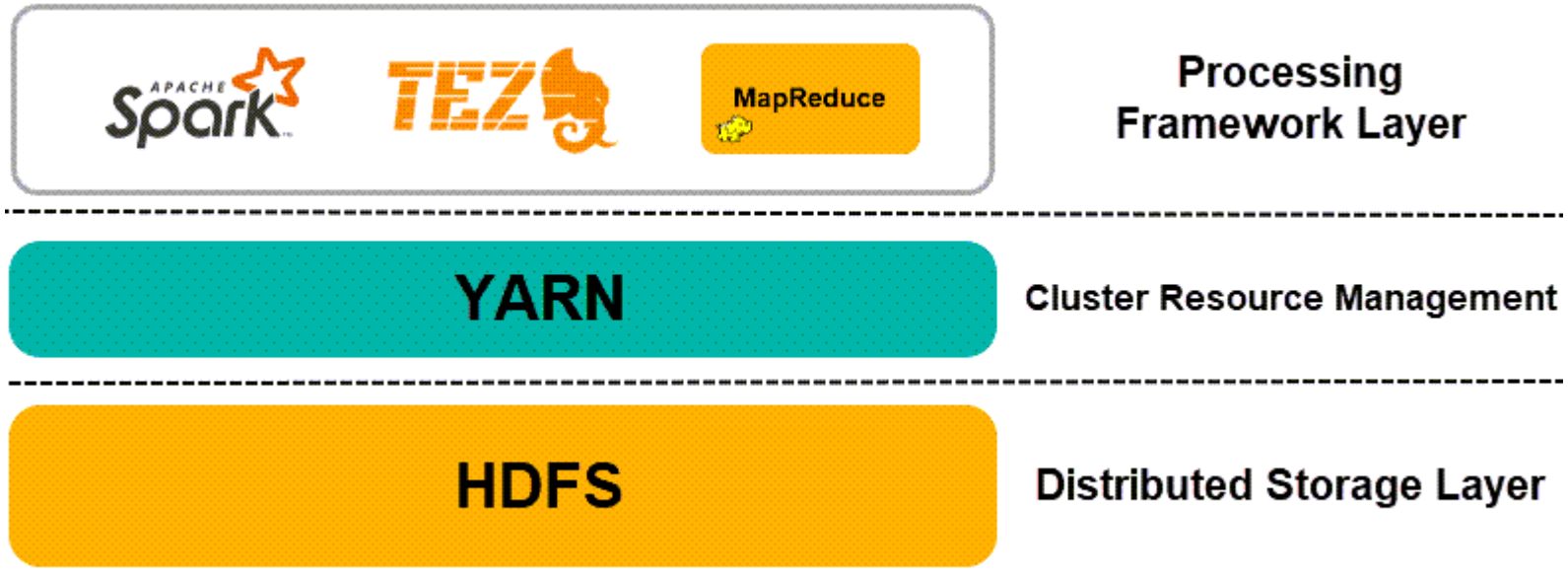
Apache Hadoop

هدوپ چیست؟

- یک چارچوب نرم افزاری است
- سیستم پردازشی توزیع شده
- فضای ذخیره سازی توزیع شده
- پردازش داده های حجیم
- ذخیره سازی داده های حجیم

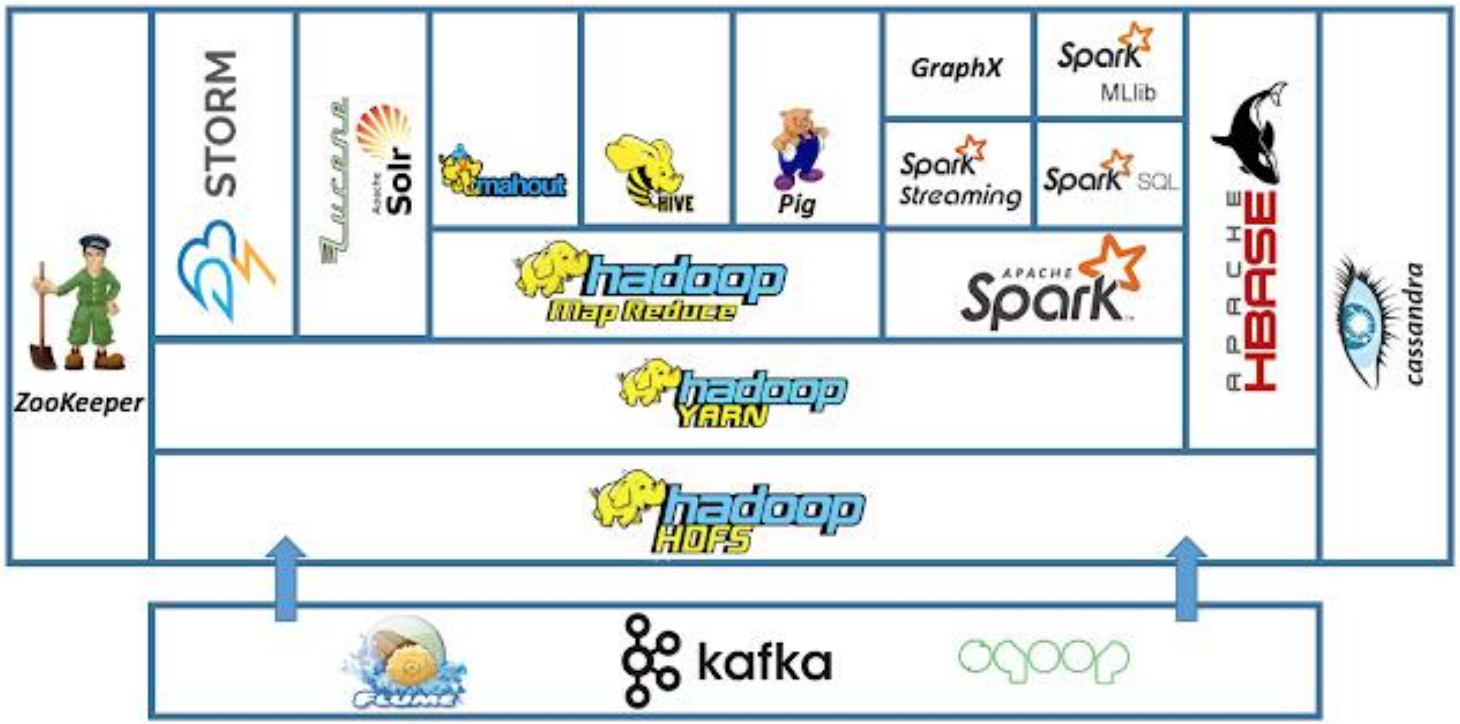
معماری Hadoop

لایه‌های اصلی هدوپ



معماری Hadoop

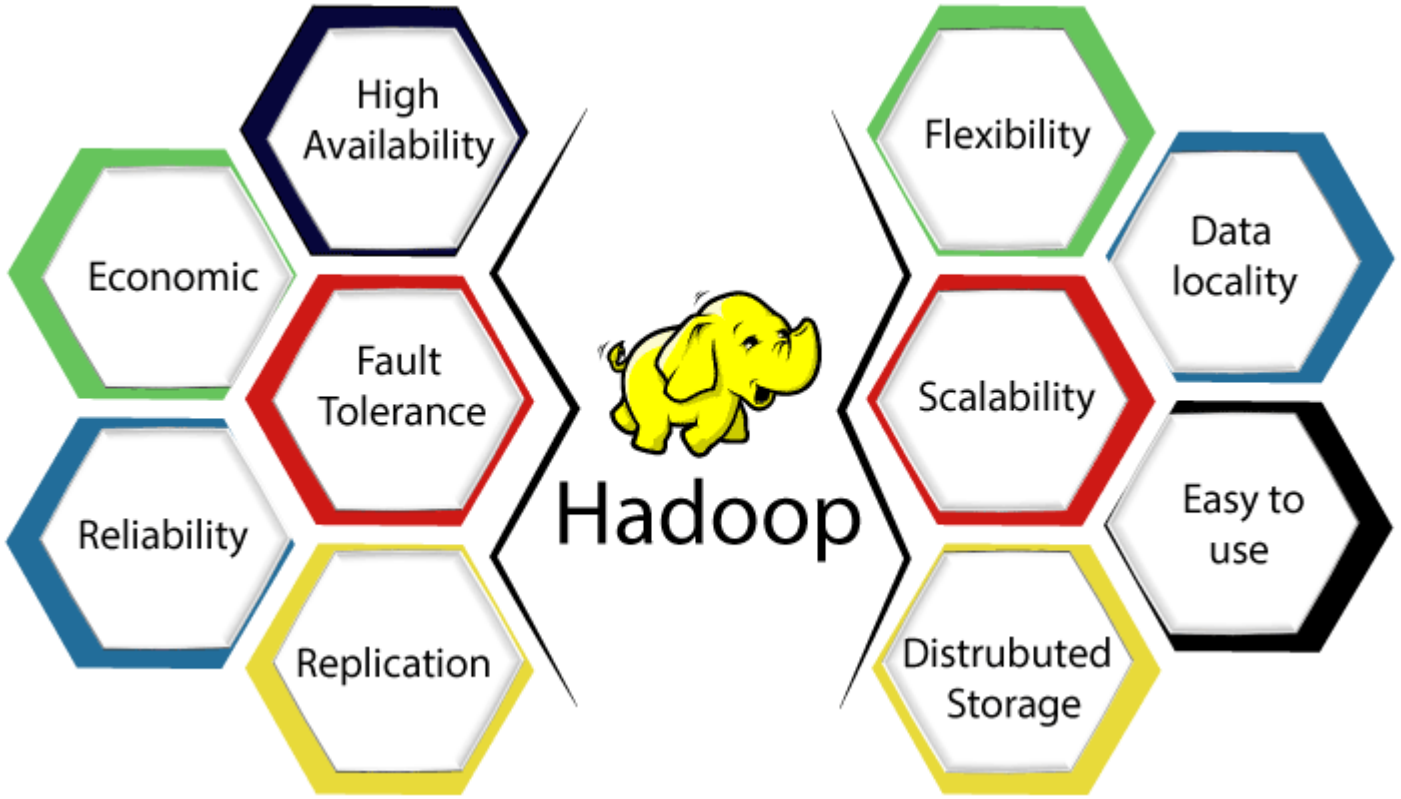
لایه‌های اصلی هدوپ



معماری Hadoop

لایه‌های اصلی هدوپ

- خصوصیات هدوپ

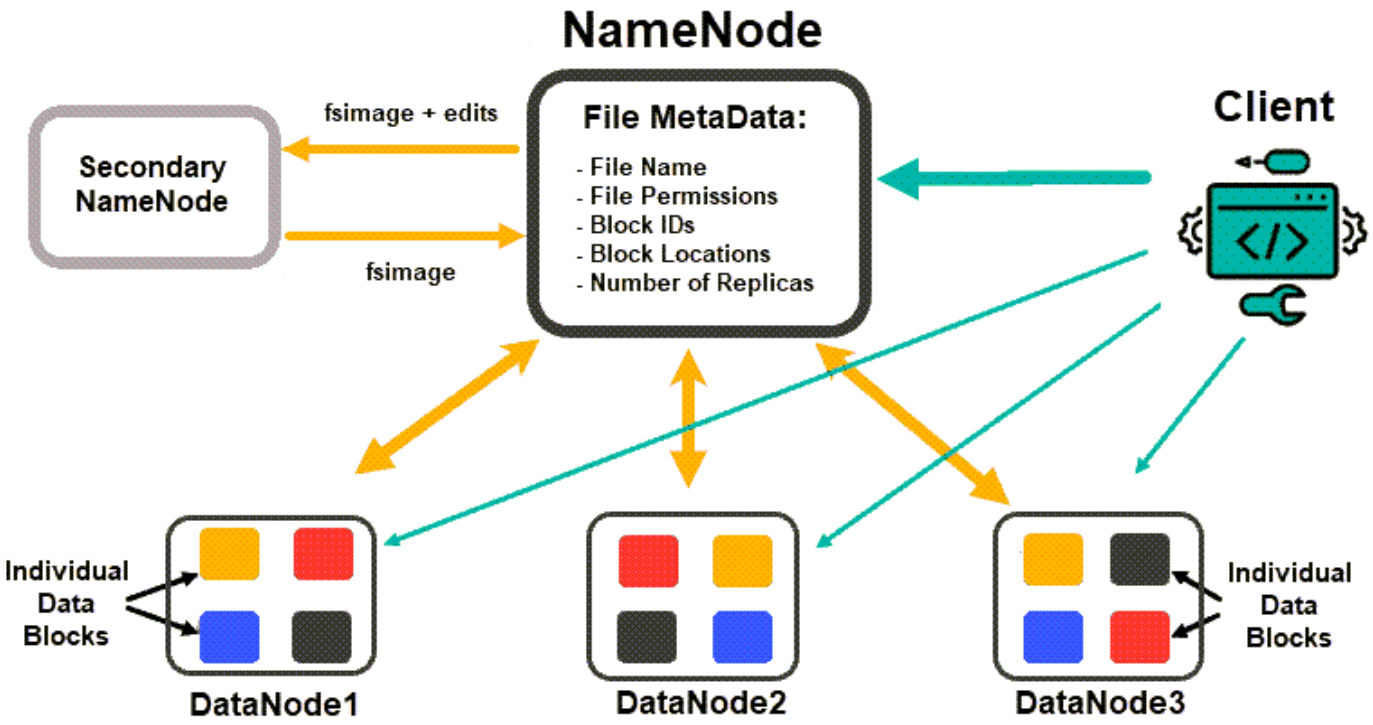


معماری Hadoop

لایه‌های اصلی هدوپ

HDFS

- Namenode
- Secondary Namenode
- Datanode
- File Blocks
- Replication

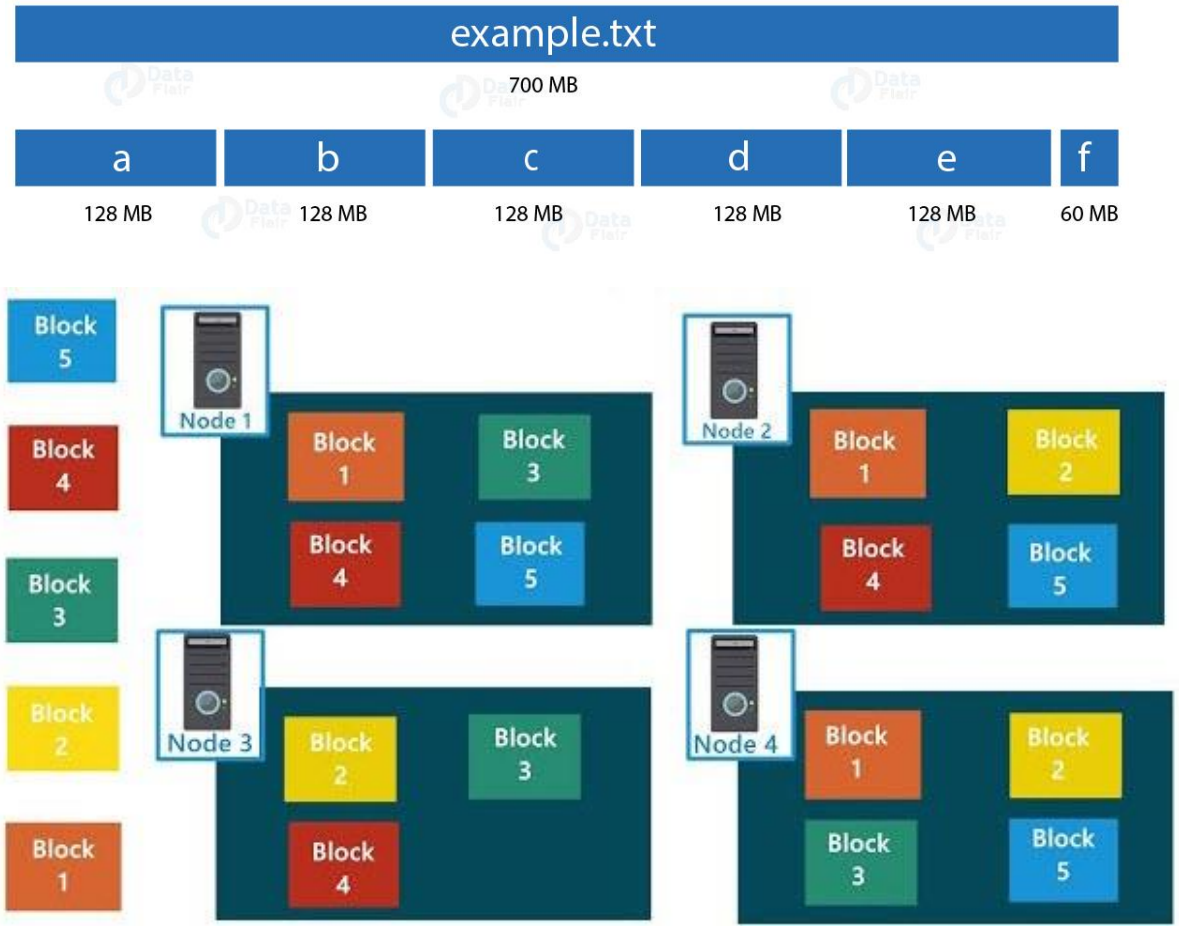


معماری Hadoop

لایه‌های اصلی هدوپ

HDFS

- Block size
- Replication

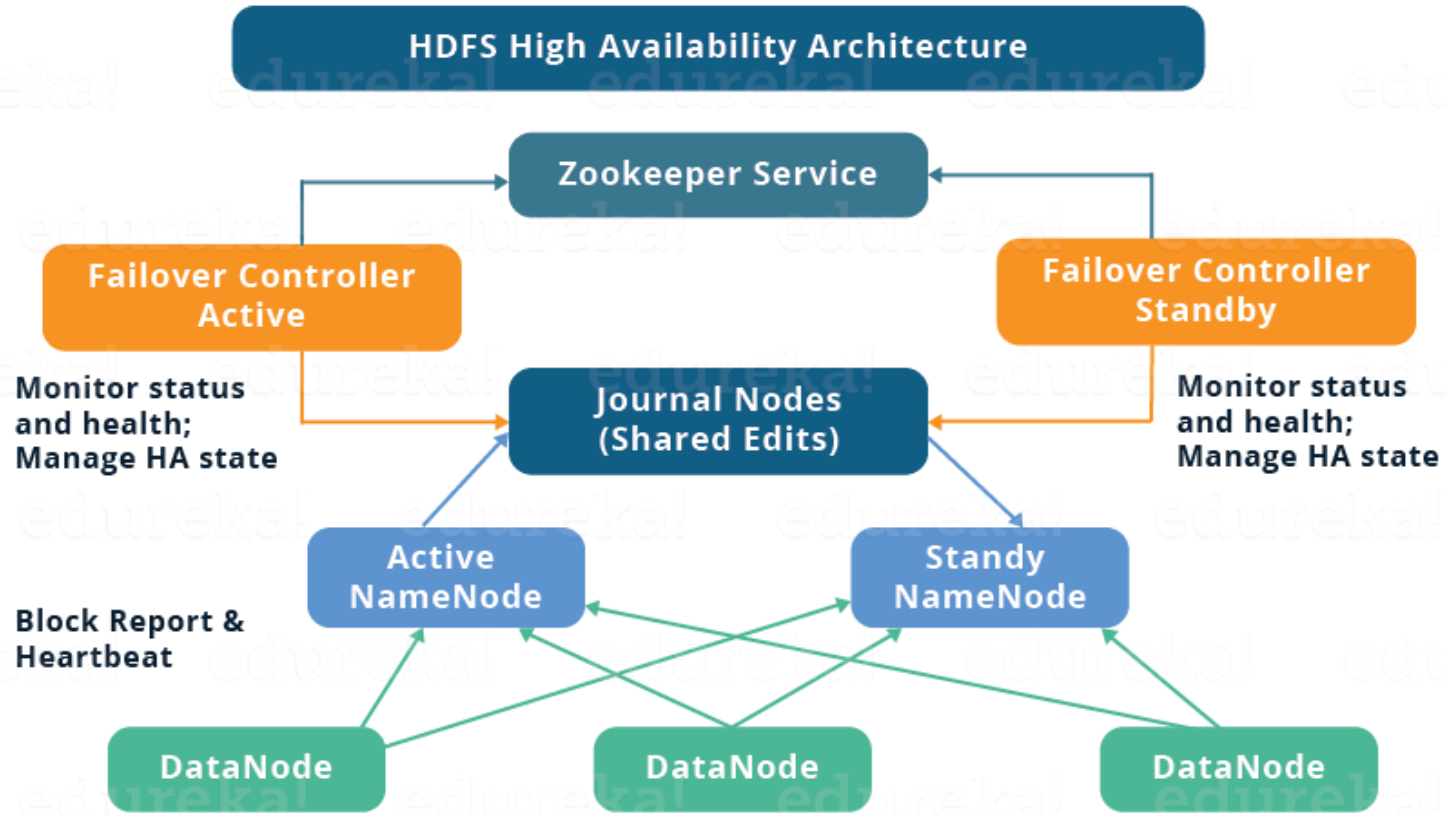


معماری Hadoop

لایه‌های اصلی هدوپ

HDFS

- Zookeeper
- Journal Nodes
- Active NN
- Standby NN



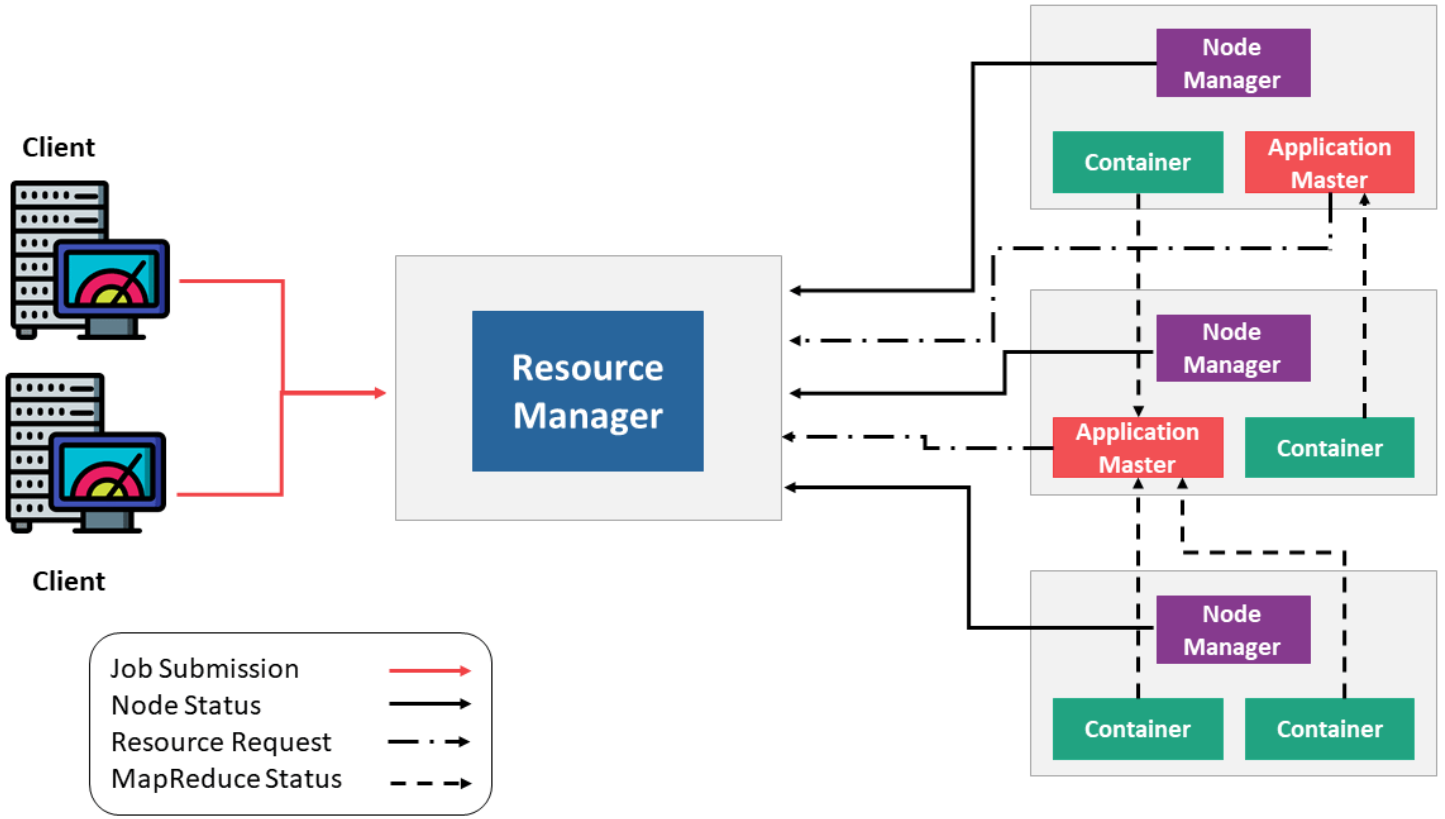
دوره جامع تکنولوژی داده

معماری Hadoop

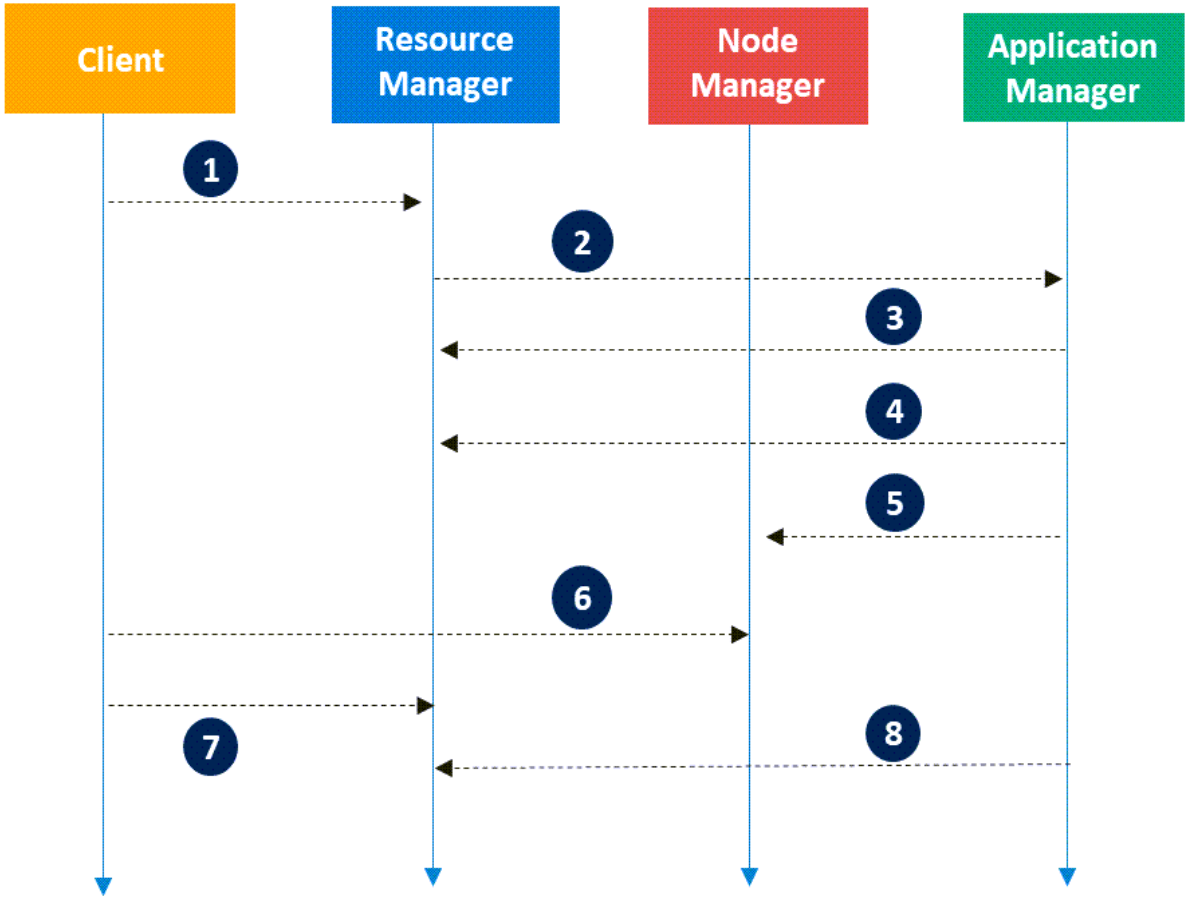
لایه‌های اصلی هدوپ

Yarn

- Resource Manager
- Node Manager
- Application Master
- Container



معماری Hadoop



لایه‌های اصلی هدوپ

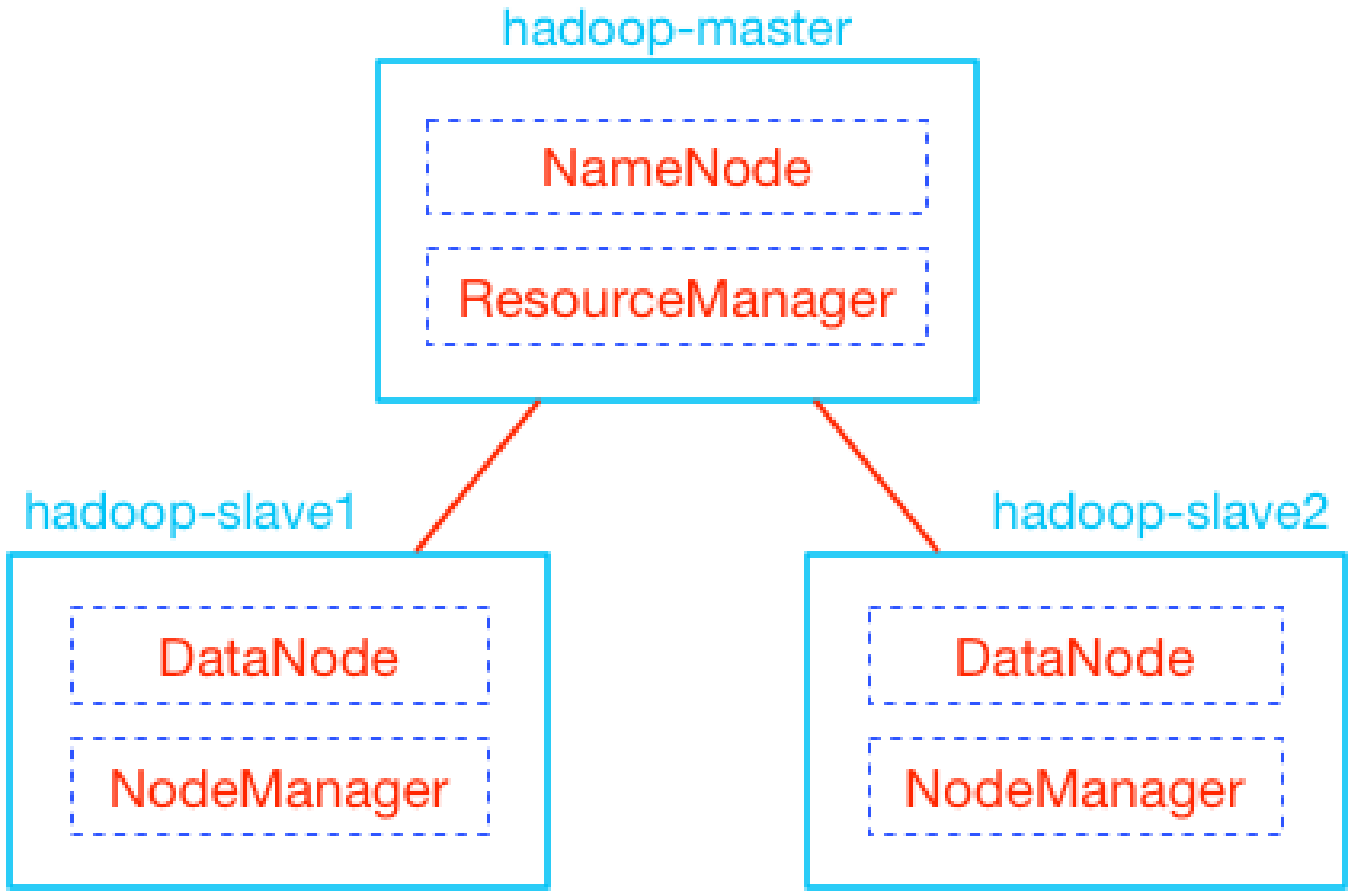
Yarn

1. اجرای برنامه توسط Client
2. درخواست container از RM برای ایجاد AM
3. رجیستر شدن AM در RM
4. درخواست Containers از AM به RM
5. درخواست اجرای container از AM به NM
6. اجرای کد در containers
7. مانیتوریگ برنامه توسط کلاینت
8. Unregister AM

معماری Hadoop

لایه‌های اصلی هدوپ

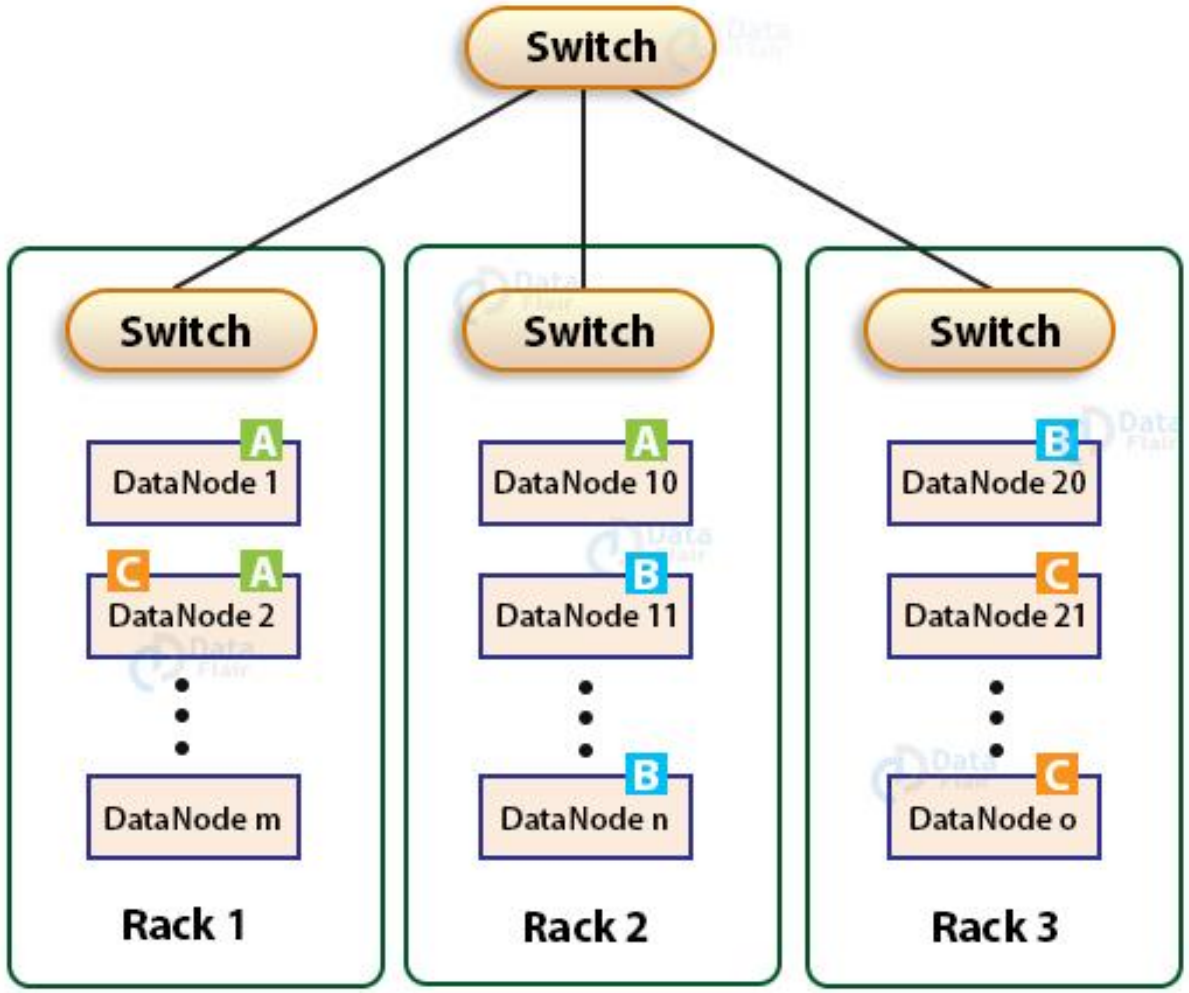
Data locality



معماری Hadoop

لایه‌های اصلی هدوپ

Rack Awareness



معماری Hadoop

لایه‌های اصلی هدوپ

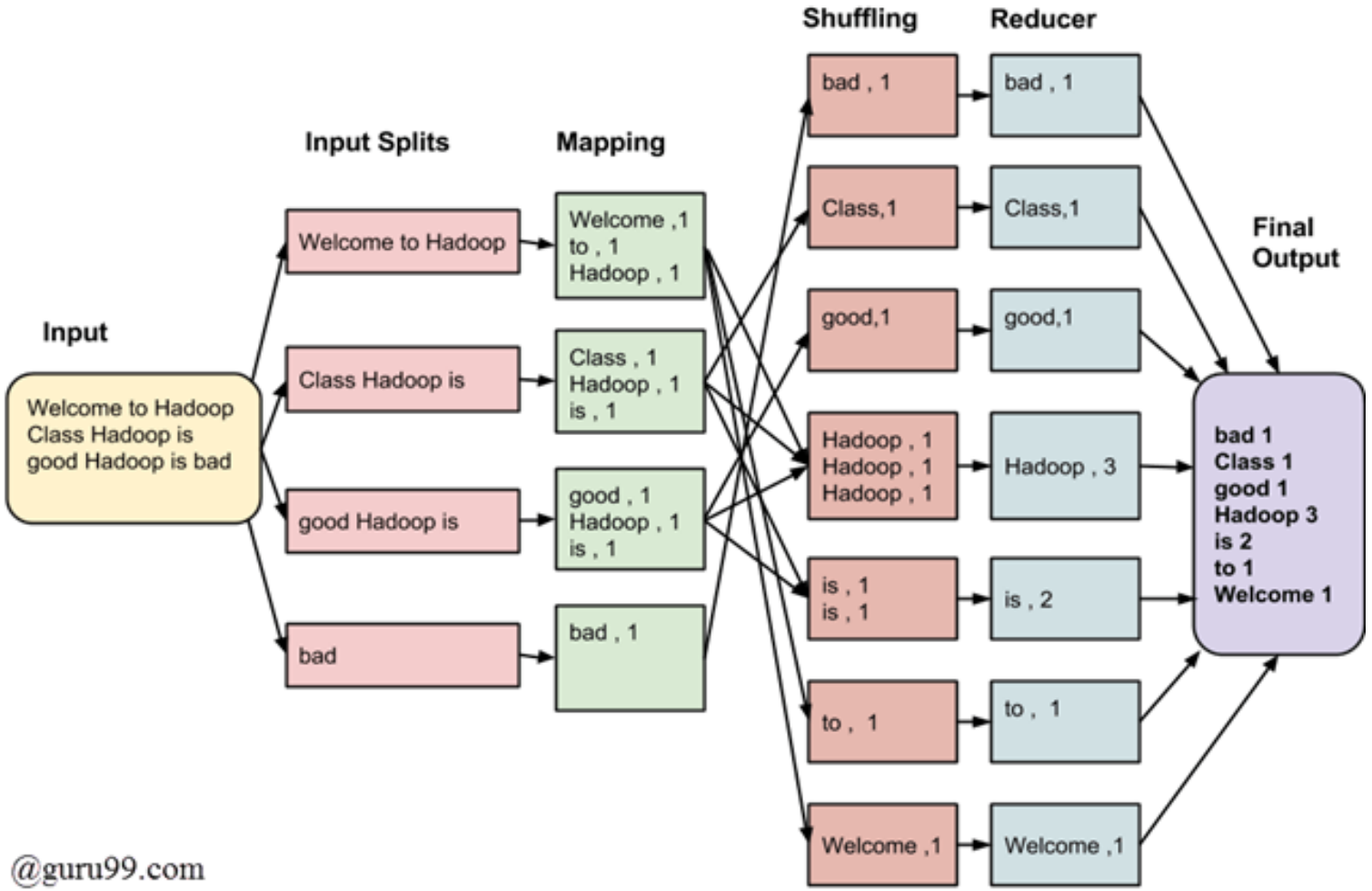
نصب و راه اندازی کلاستر هدوپ

نمایش کافیگ‌های

https://www.linode.com/docs/guides/how-to-install-and-set-up-hadoop-cluster/

معماری Hadoop

MapReduce



@guru99.com

کارعملی Hadoop

MapReduce

```
2   import ...
14
15  public class WordCount {
16
17      public static class TokenizerMapper
18              extends Mapper<Object, Text, Text, IntWritable>{...}
32
33      public static class IntSumReducer
34              extends Reducer<Text,IntWritable,Text,IntWritable> {...}
48
49      public static void main(String[] args) throws Exception {...}
62  }
```

کارعملی Hadoop

MapReduce

```
17  public static class TokenizerMapper
18          extends Mapper<Object, Text, Text, IntWritable>{
19
20      private final static IntWritable one = new IntWritable( value: 1);
21      private Text word = new Text();
22
23  @   public void map(Object key, Text value, Context context
24      ) throws IOException, InterruptedException {
25          StringTokenizer itr = new StringTokenizer(value.toString());
26          while (itr.hasMoreTokens()) {
27              word.set(itr.nextToken());
28              context.write(word, one);
29          }
30      }
31  }
```

کارعملی Hadoop

MapReduce

```
33      public static class IntSumReducer
34              extends Reducer<Text,IntWritable,Text,IntWritable> {
35          private IntWritable result = new IntWritable();
36
37          public void reduce(Text key, Iterable<IntWritable> values,
38                             Context context
39          ) throws IOException, InterruptedException {
40              int sum = 0;
41              for (IntWritable val : values) {
42                  sum += val.get();
43              }
44              result.set(sum);
45              context.write(key, result);
46          }
47      }
```

MapReduce

```
49  ▶ @    public static void main(String[] args) throws Exception {
50             Configuration conf = new Configuration();
51             Job job = Job.getInstance(conf, jobName: "word count");
52             job.setJarByClass(WordCount.class);
53             job.setMapperClass(TokenizerMapper.class);
54             job.setCombinerClass(IntSumReducer.class);
55             job.setReducerClass(IntSumReducer.class);
56             job.setOutputKeyClass(Text.class);
57             job.setOutputValueClass(IntWritable.class);
58             FileInputFormat.addInputPath(job, new Path(args[0]));
59             FileOutputFormat.setOutputPath(job, new Path(args[1]));
60             System.exit(job.waitForCompletion( verbose: true) ? 0 : 1);
61         }
```

کارعملی Hadoop

MapReduce

- ایجاد یک فایل ورودی
- ارسال فایل به Hadoop

```
[mdalirian@dahdpfpsbpp0105 ~]$ hdfs dfs -put mapredtest.txt
[mdalirian@dahdpfpsbpp0105 ~]$ hdfs dfs -ls /user/mdalirian/mapredtest.txt
-rw-r--r--   2 mdalirian supergroup        107 2021-09-11 04:14 /user/mdalirian/mapredtest.txt
[mdalirian@dahdpfpsbpp0105 ~]$
```

MapReduce

- اجرای برنامه WordCount
- مراحل اجرای برنامه بصورت عملی نشان داده می‌شود.

```
[mdaliriaan@dahdpfpsbpp0105 jars]$ yarn  jar job.jar  inputDir outputDir
```

کارعملی Hadoop

```
[mdalirian@dahdpfpsbpp0105 ~]$ hdfs dfs
Usage: hadoop fs [generic options]
        [-appendToFile <localsrc> ... <dst>]
        [-cat [-ignoreCrc] <src> ...]
        [-checksum <src> ...]
        [-chgrp [-R] GROUP PATH...]
        [-chmod [-R] <MODE[,MODE]... | OCTALMODE> PATH...]
        [-chown [-R] [OWNER][:[GROUP]] PATH...]
        [-copyFromLocal [-f] [-p] [-l] [-d] [-t <thread count>] <localsrc> ... <dst>]
        [-copyToLocal [-f] [-p] [-ignoreCrc] [-crc] <src> ... <localdst>]
        [-count [-q] [-h] [-v] [-t [<storage type>]] [-u] [-x] [-e] <path> ...]
        [-cp [-f] [-p | -p[topax]] [-d] <src> ... <dst>]
        [-createSnapshot <snapshotDir> [<snapshotName>]]
        [-deleteSnapshot <snapshotDir> <snapshotName>]
        [-df [-h] [<path> ...]]
        [-du [-s] [-h] [-v] [-x] <path> ...]
        [-expunge]
        [-find <path> ... <expression> ...]
        [-get [-f] [-p] [-ignoreCrc] [-crc] <src> ... <localdst>]
        [-getfacl [-R] <path>]
        [-getfattr [-R] {-n name | -d} [-e en] <path>]
        [-getmerge [-nl] [-skip-empty-file] <src> <localdst>]
        [-help [cmd ...]]
        [-ls [-C] [-d] [-h] [-q] [-R] [-t] [-S] [-r] [-u] [-e] [<path> ...]]
        [-mkdir [-p] <path> ...]
        [-moveFromLocal <localsrc> ... <dst>]
        [-moveToLocal <src> <localdst>]
        [-mv <src> ... <dst>]
        [-put [-f] [-p] [-l] [-d] <localsrc> ... <dst>]
        [-renameSnapshot <snapshotDir> <oldName> <newName>]
        [-rm [-f] [-r|-R] [-skipTrash] [-safely] <src> ...]
        [-rmdir [--ignore-fail-on-non-empty] <dir> ...]
        [-setfacl [-R] [{-b|-k} {-m|-x <acl_spec>} <path>]|[--set <acl_spec> <path>]]
        [-setfattr {-n name [-v value] | -x name} <path>]
        [-setrep [-R] [-w] <rep> <path> ...]
        [-stat [format] <path> ...]
        [-tail [-f] <file>]
        [-test -[defsz] <path>]
        [-text [-ignoreCrc] <src> ...]
        [-touch [-a] [-m] [-t TIMESTAMP ] [-c] <path> ...]
```

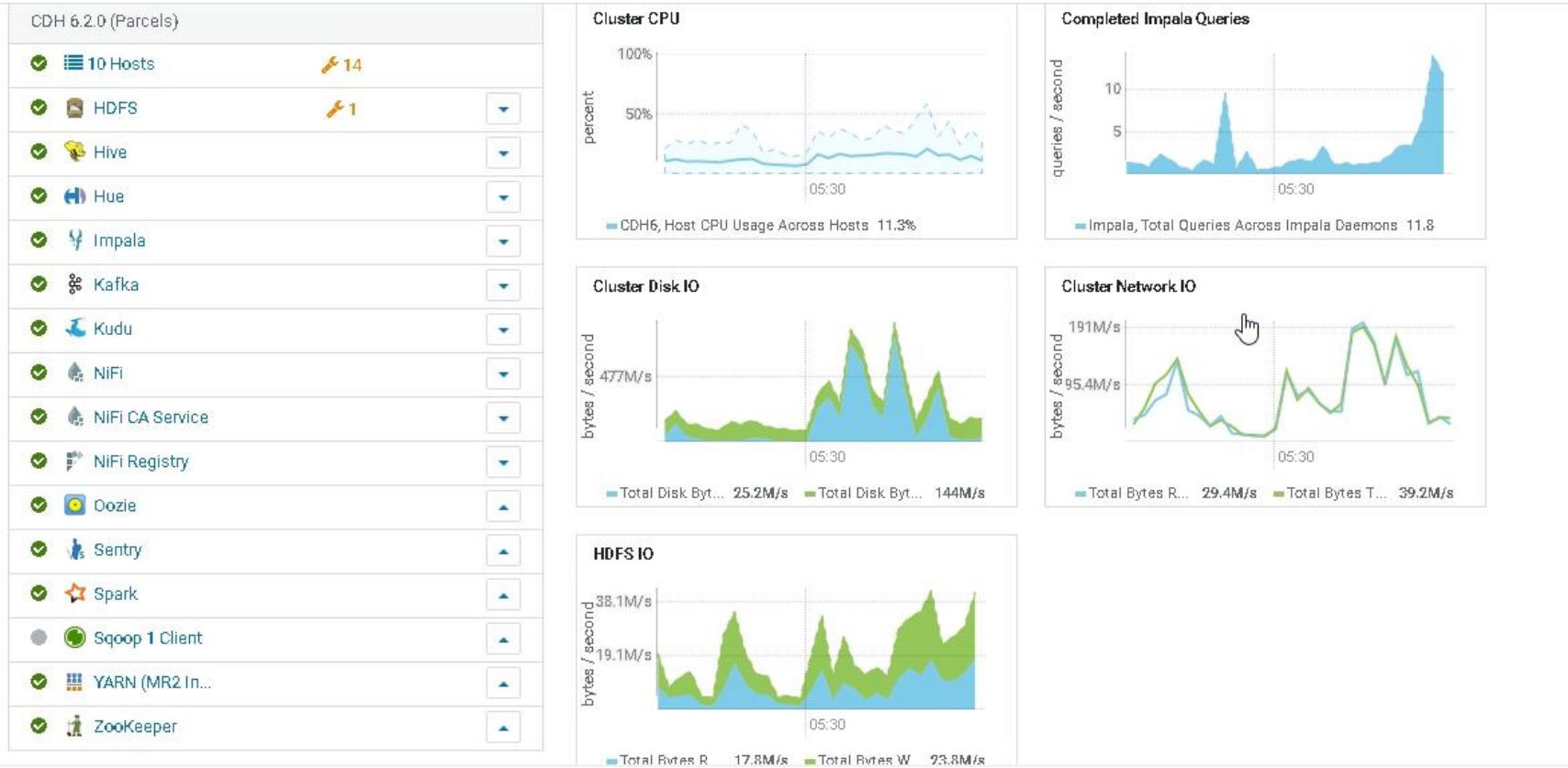
Hadoop Commands

- مسیرهای مختلف در هدوپ
- سطح دسترسی
- اجرای دستورات

کارعملی Hadoop

نمایش محیط هدوپ

- معرفی Cloudera
- معرفی و نمایش ابزارها



Apache Spark

Spark چیست؟

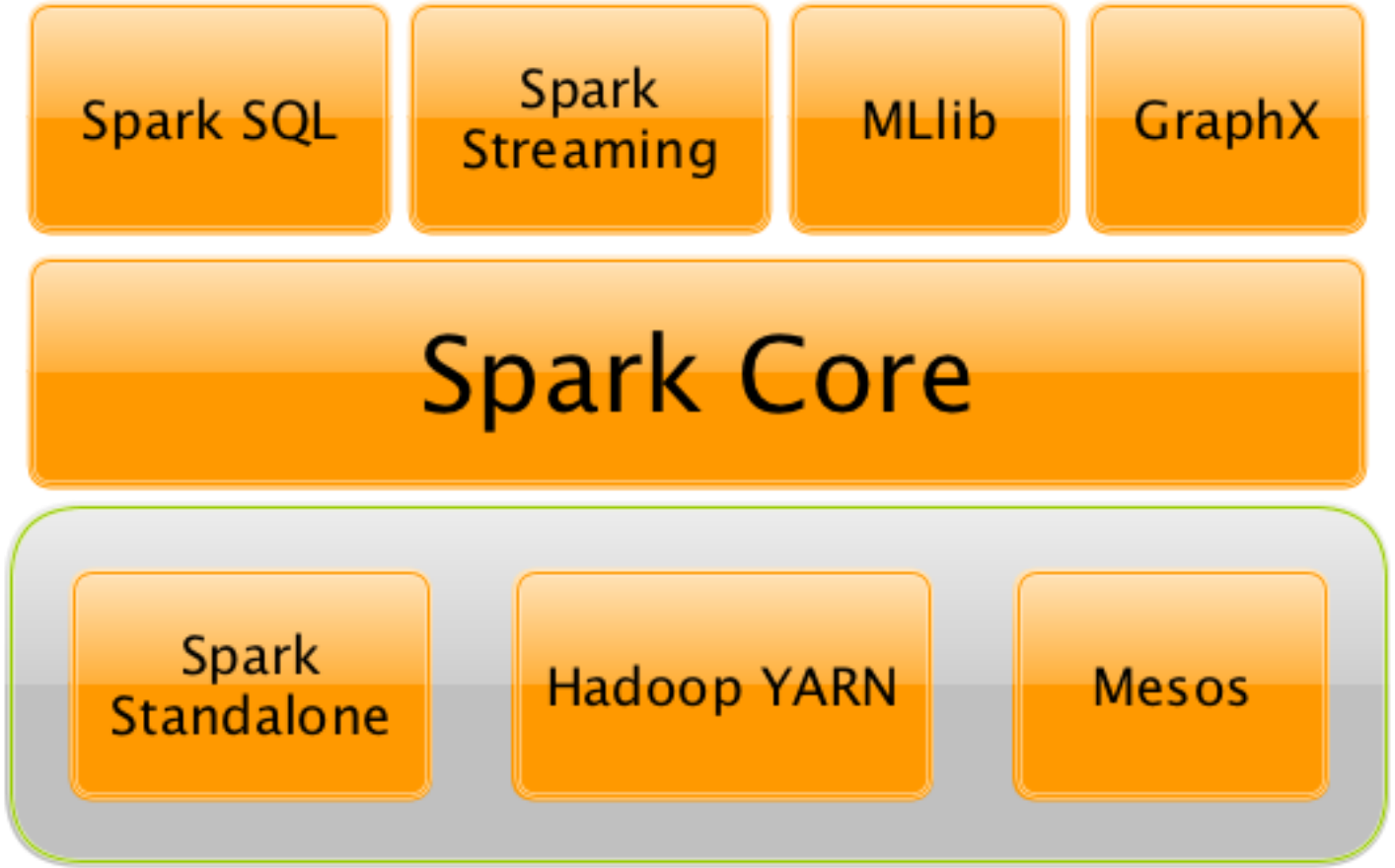
- یک چارچوب نرم افزاری است
- سیستم پردازشی توزیع شده
- استفاده از حافظه داخلی جهت پردازش سریع
- پردازش دسته ای
- پردازش جریان داده
- اجرای پرس وجوهای تعاملی
- یادگیری ماشین
- پردازش گراف

معماری Spark

خصوصیات Spark

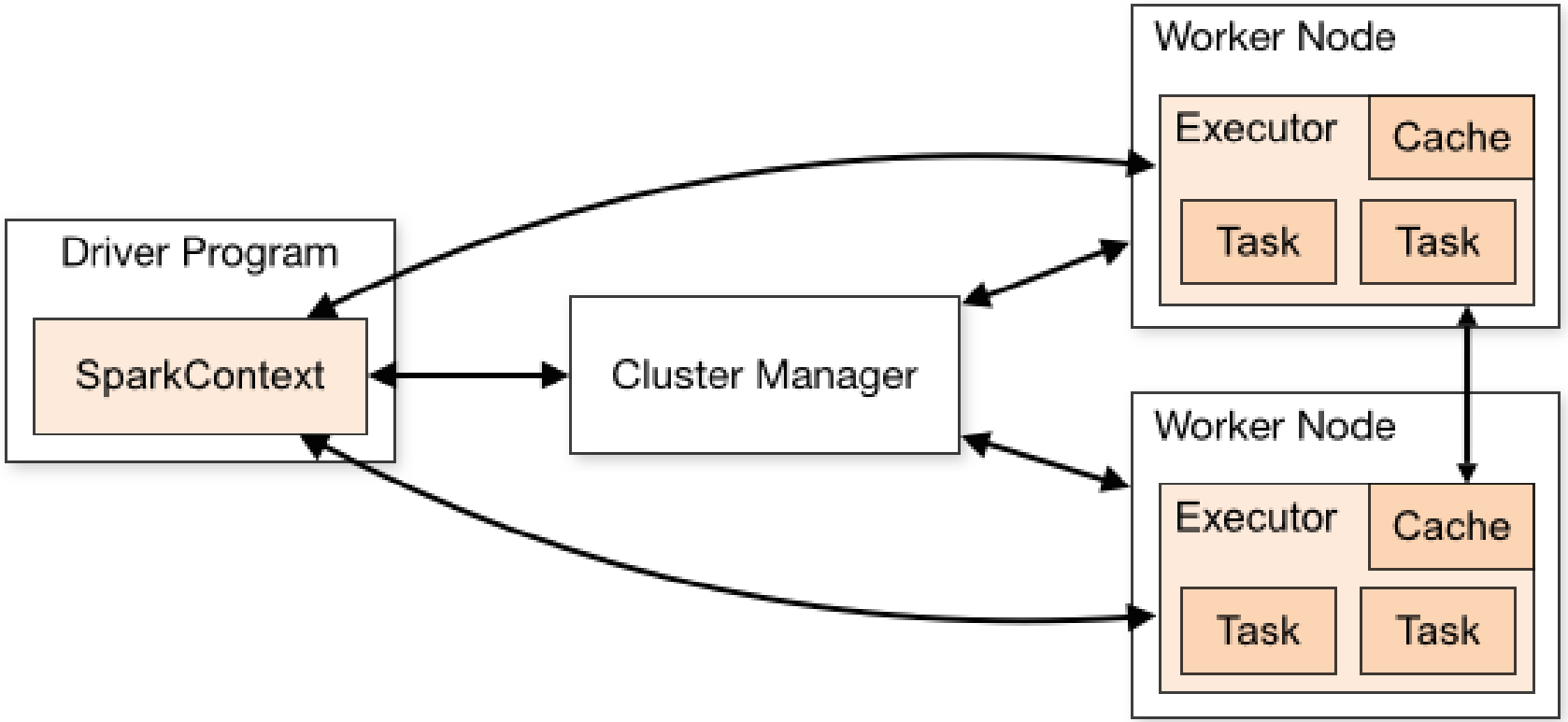
معماری Spark

معماری spark



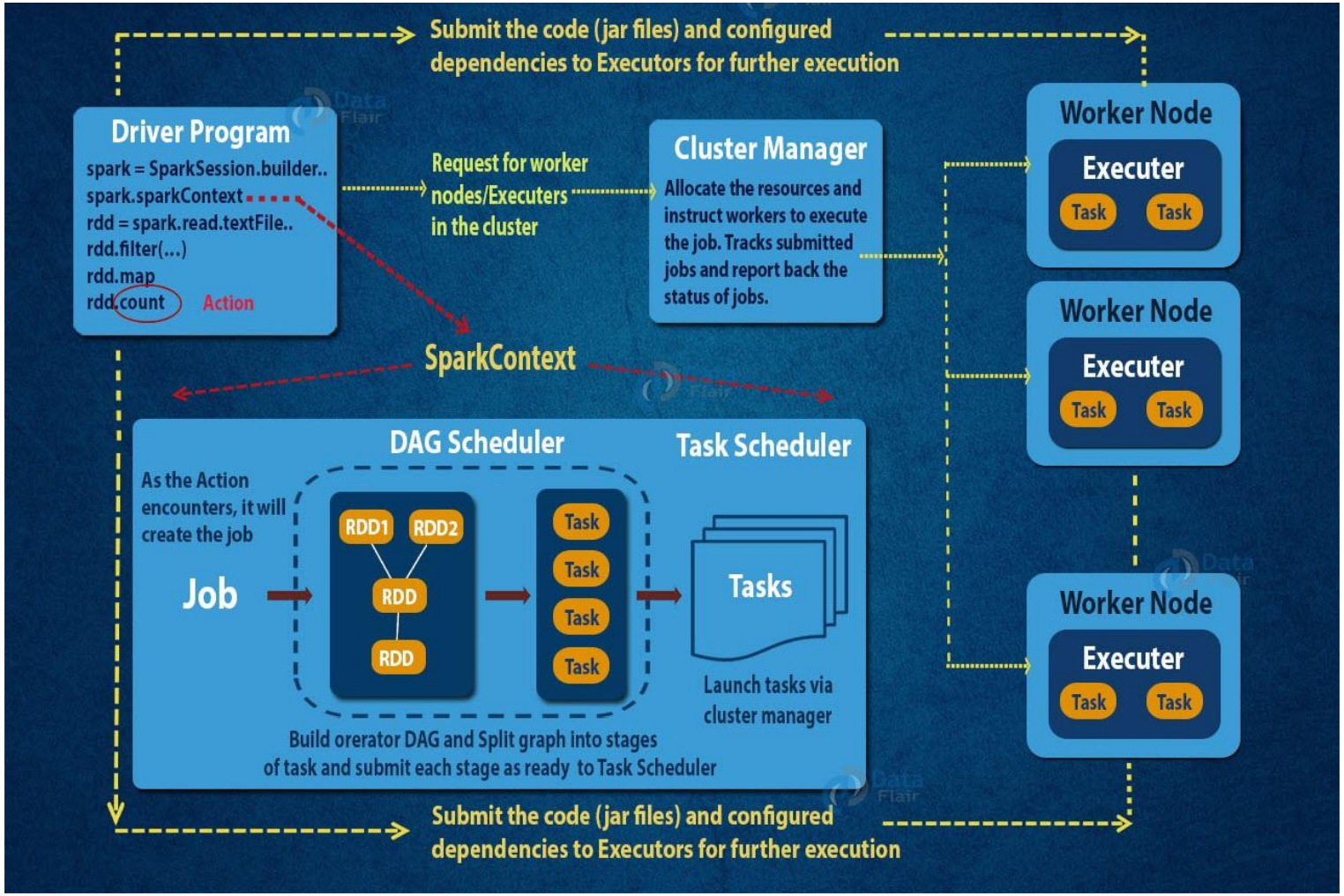
معماری Spark

معماری spark

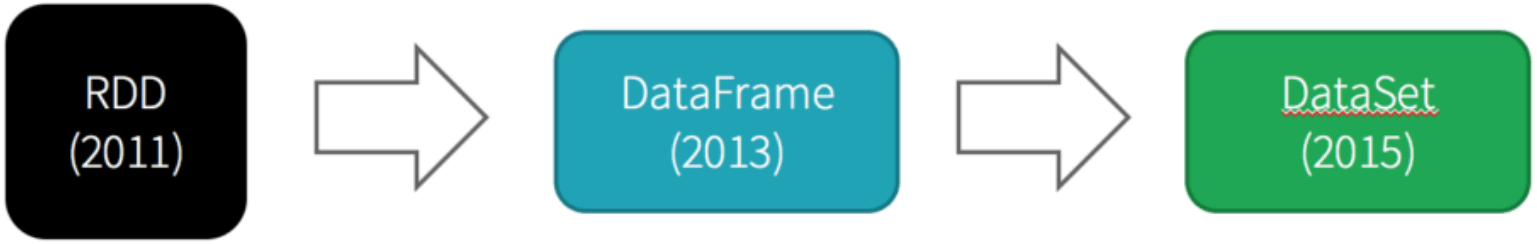


معماری Spark

معماری spark



History of Spark APIs



Distribute collection of JVM objects

Functional Operators (map, filter, etc.)

Distribute collection of Row objects

Expression-based operations and UDFs

Logical plans and optimizer

Fast/efficient internal representations

Internally rows, externally JVM objects

Almost the "Best of both worlds": type safe + fast

But slower than DF
Not as good for interactive analysis, especially Python

Spark API

- RDD
- Dataframes
- Datasets

databricks

RESEARCH

RDDs Operations

- Transformation
- Action

Transformation:

- Takes RDD as input and produce one or many RDDs
- Lazy operations
- Auto optimization applied, Ex :pipeline
- Ex: Map, Filter, Distinct, etc.

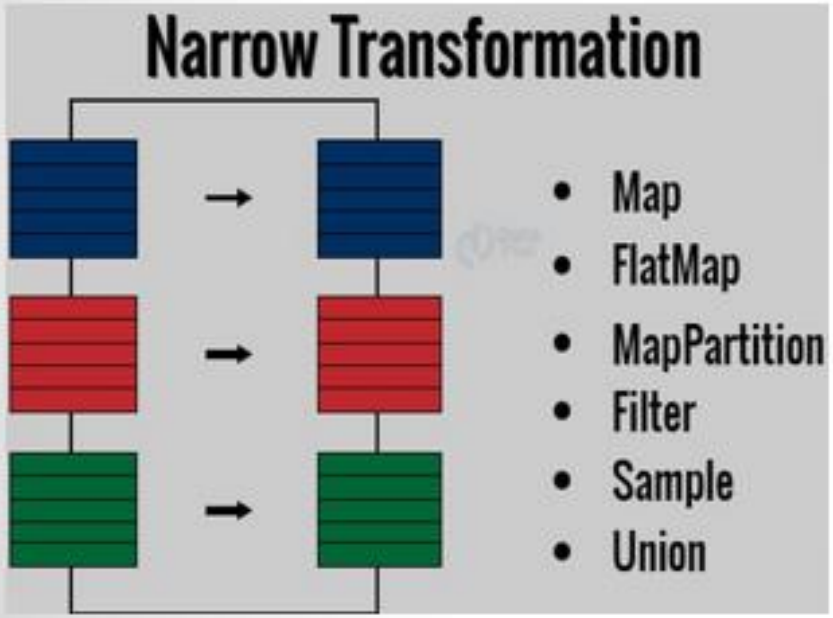
Actions:

- Triggers execution
- Produce non-RDD values
- Ex: First(), take(), reduce(), etc.

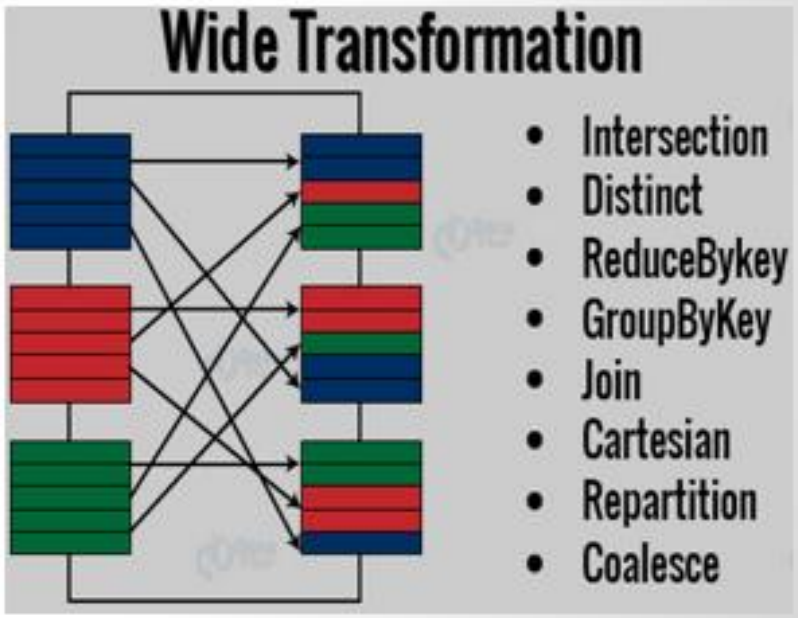
RDDs Operations

- Transformation

1) Narrow Transformation



2) Wide Transformation



Spark Examples

```scala
import org.apache.spark.sql.SparkSession

object WordCount extends App {
  val spark = SparkSession.builder().appName( name = "wordCount")
    .master( master = "local[*]").getOrCreate()
  var file = spark.sparkContext.textFile( path = "testfile.txt")
  val counts = file.flatMap(line => line.split( regex = " ")).map(word => (word, 1)).reduceByKey(_+_)
  counts.foreach(rdd=>println(rdd._1+": "+rdd._2))
  spark.close()
}
```

Spark Examples

```scala
object RDDtoDF extends App {
  val spark = SparkSession.builder().appName( name = "RDDtoDF")
    .master( master = "local[*]").getOrCreate()
  val sc = spark.sparkContext

  import spark.implicits._

  val simpleData = Seq(
    ("James", "Sales", "NY", 90000, 34, 10000),
    ("Michael", "Sales", "NY", 86000, 56, 20000),
    ("Kumar", "Marketing", "NY", 91000, 50, 21000)
  )
  val rdd = sc.parallelize(simpleData)
  val df = rdd.toDF( colNames = "employee_name", "department", "state", "salary", "age", "bonus")
  df.show();
  spark.close()
}
```

Spark Examples

```scala
import org.apache.spark.sql.SparkSession

object ReadCSVFile extends App {
  val spark:SparkSession = SparkSession.builder().master( master = "local[1]")
    .appName( name = "ReadCSVFile").getOrCreate()
  spark.sparkContext.setLogLevel("ERROR")
  val rddFromFile = spark.sparkContext.textFile( path = "E:\\IntellijProjects\\firstScalaProject\\file1.csv")

  val rdd = rddFromFile.map(f=>{
    f.split( regex = ",")
  })

  rdd.foreach(f=>{
    println("Col1:"+f(0)+",Col2:"+f(1))
  })
  spark.close()
}
```

Spark Examples

مثال‌های بیشتر که بصورت عملی نوشته و اجرا خواهد شد:

- Spark Repartition() vs Coalesce()
- Spark Broadcast Variables
- Spark Accumulators
- Spark DataFrame withColumn
- Spark SQL StructType & StructField
- Spark SQL Join

Spark Examples

مثال‌های بیشتر که بصورت عملی نوشته و اجرا خواهد شد:

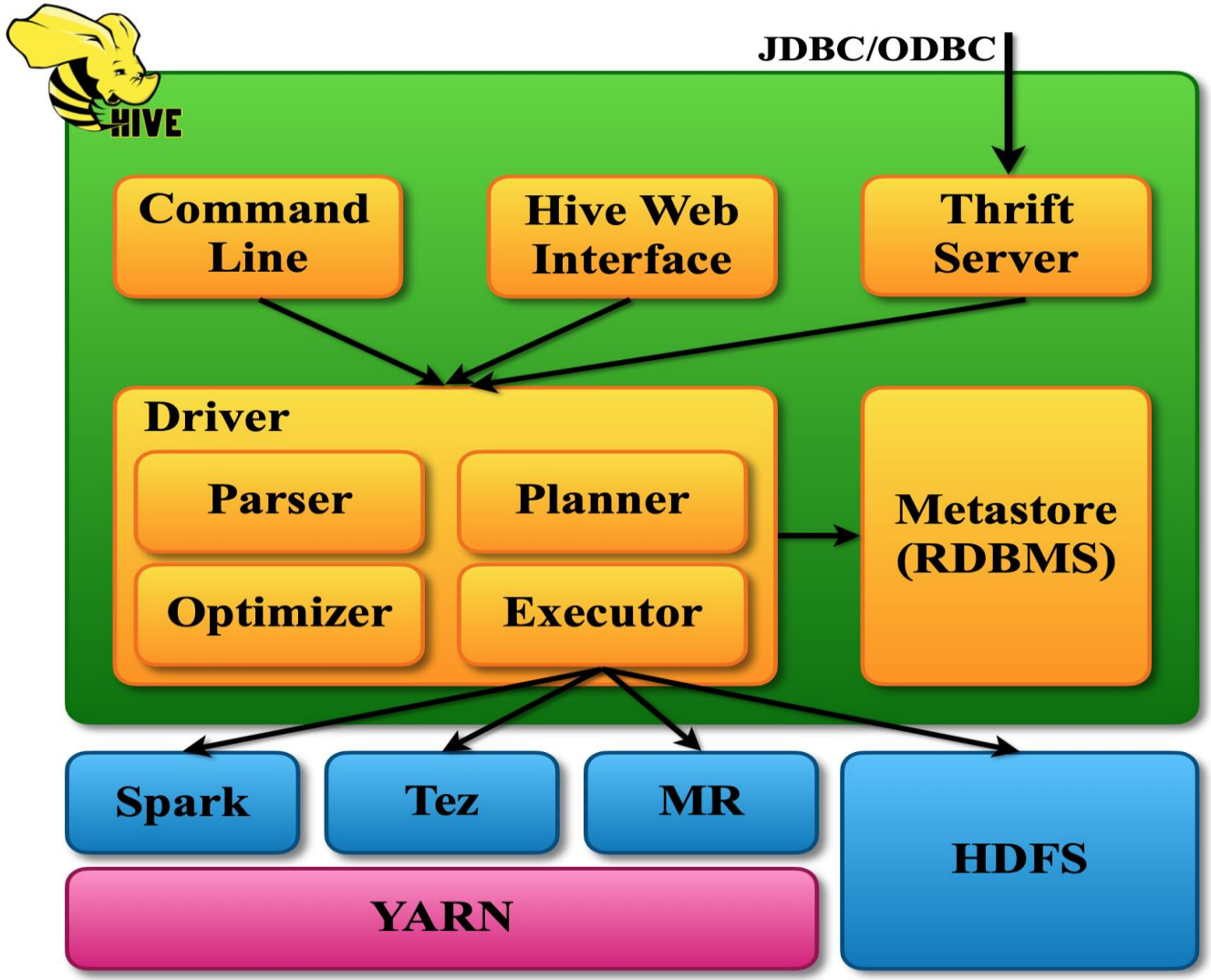
- Spark SQL UDF
- Spark SQL Aggregate Functions
- Spark Read and Write JSON file into DataFrame
- Spark Streaming with Kafka

Apache Hive

Hive چیست؟

- موتور پرس وجو مشابه SQL (SQL-like Query Engine)
- قابلیت ایجاد اسکیما برای داده‌ها
- پردازش دسته‌ای داده‌های حجیم
- بر بستر هدوپ ایجاد شده است
- بعنوان ابزار ETL استفاده می‌شود

معماری Hive



معماری Hive

- Hive Clients
- Hive Services
- Hive Storage And Computing

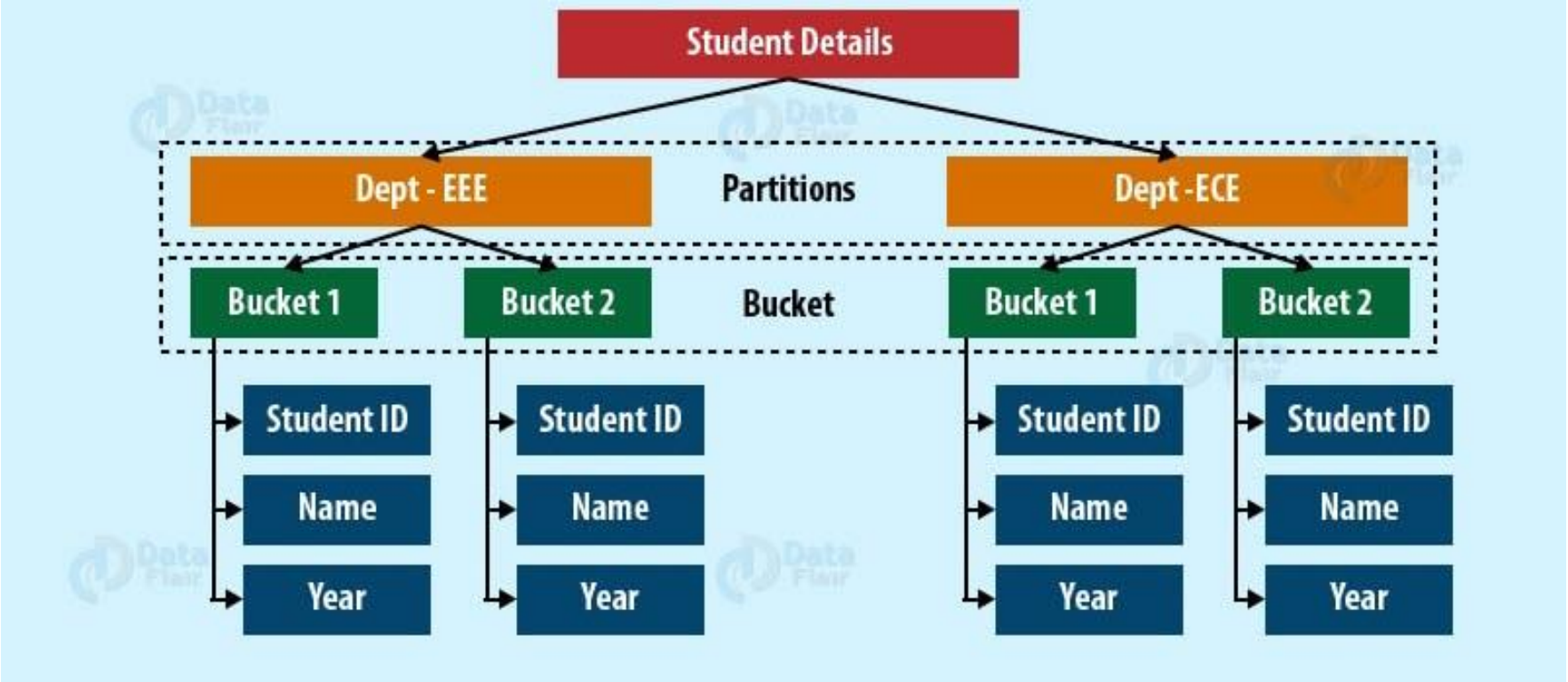
دانشگاه خاتم
Khatam University
فناپ
FANAP
دوره جامع تکنولوژی داده
RESEARCH

معماری Hive

معماری Hive

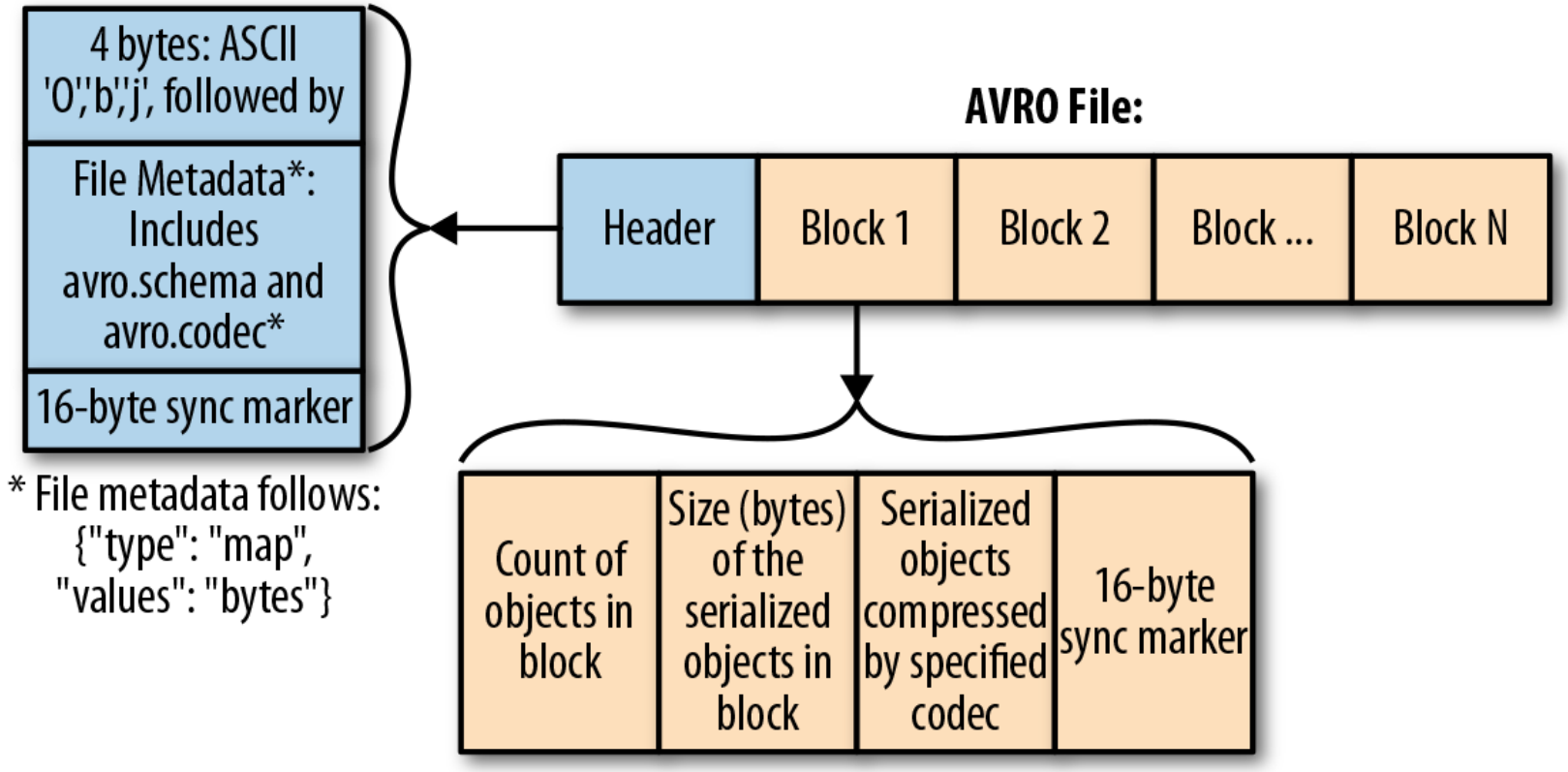
Data Model

- Database
- Tables
- Partitions
- Buckets



فرمت های ذخیره سازی

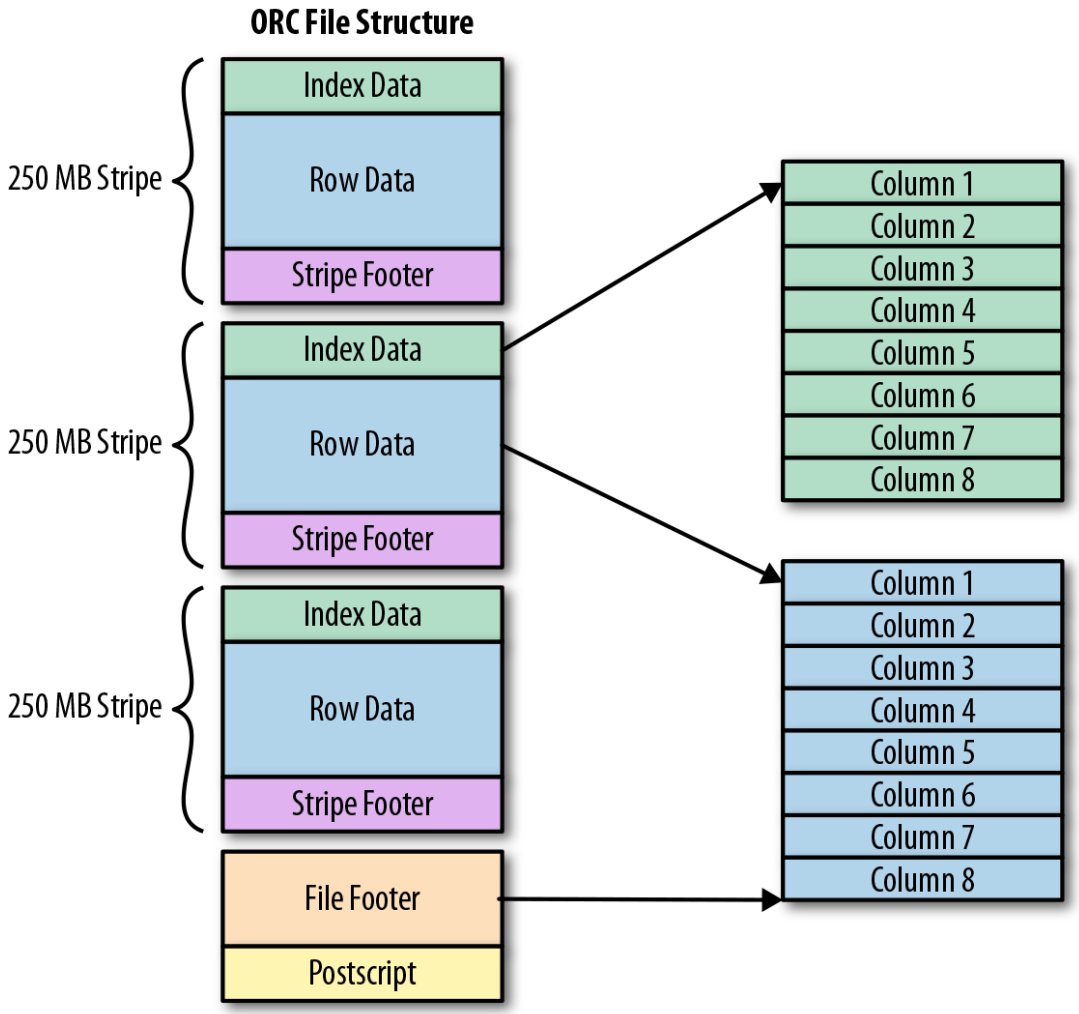
- Avro



معماری Hive

فرمت های ذخیره سازی

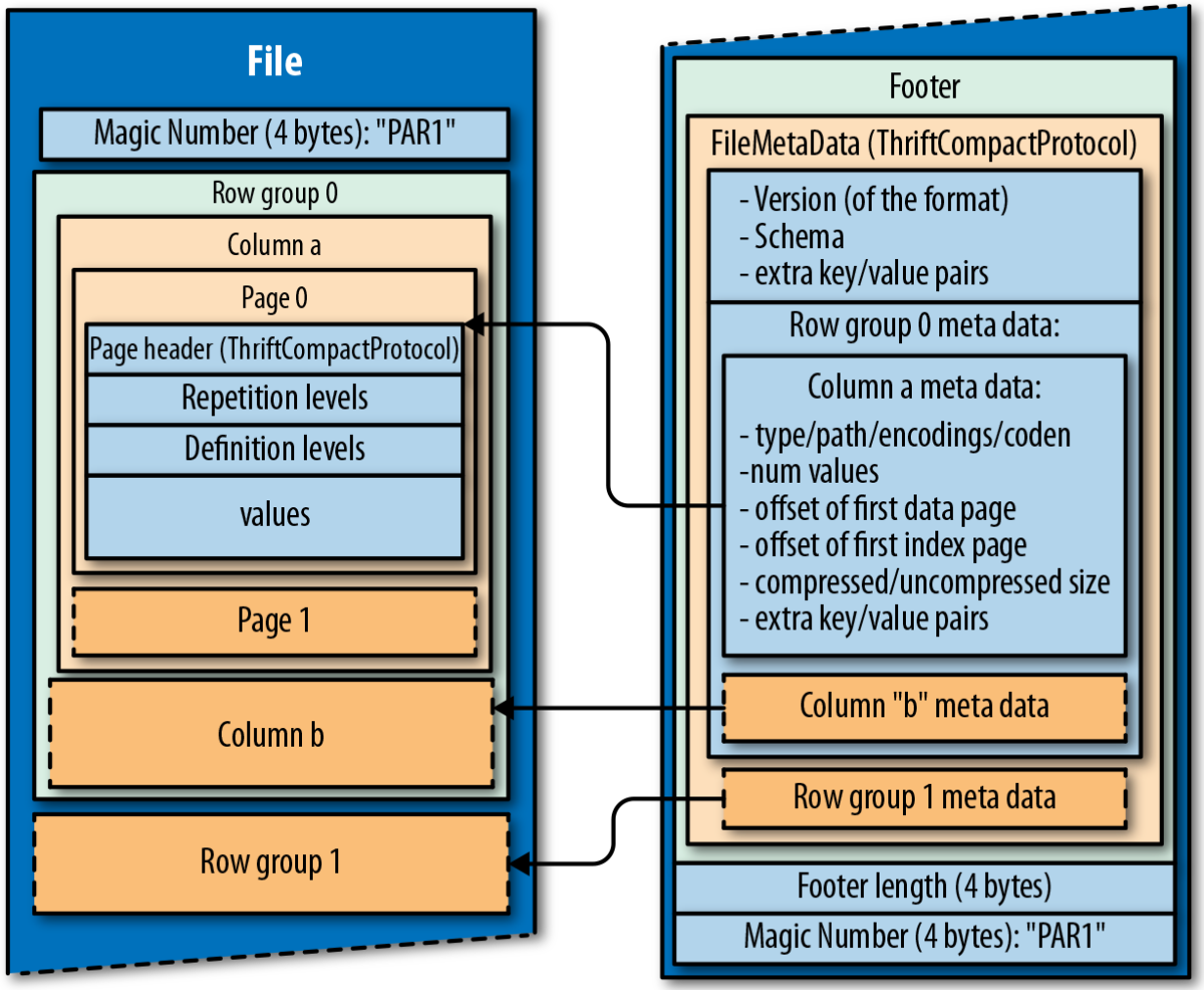
- ORC (Optimized Row Column)



معماری Hive

فرمت های ذخیره سازی

- Parquet



معماری Hive

فرمت های ذخیره سازی

	ORC	Parquet	Avro
Row or column	Column	Column	Row
Compression	Great	Great	Good
Speedup (compared to text file)	10-100x	10-100x	10x
Schema evolution	Good	Better	Best
Platforms	Hive, Spark, Presto	Hive, Spark, Presto	Hive, Spark
Splittability	Best	Best	Better
File statistics	Yes	Yes	No
Indexes	Yes	Yes	No
Bloom filters	Yes	No	No

Cloudera Impala

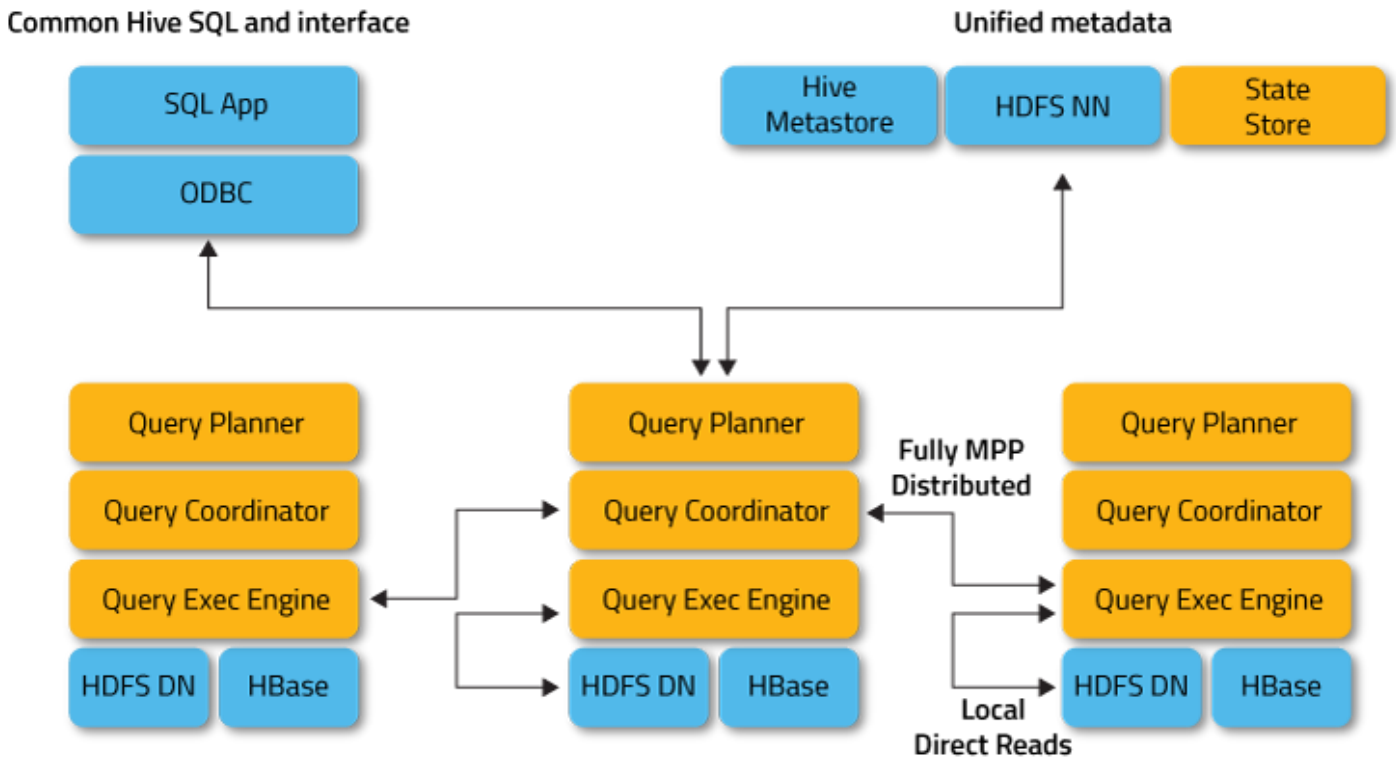
Impala چیست؟

- موتور پرس‌وجو توزیع شده SQL
- پردازش موازی انبوه (MPP)
- مشابه hive با سیستم پردازش مجزا

معماری Impala

معماری Impala

- Impala Daemon
- Impala StateStore
- Impala Metadata & MetaStore



کارعملی

- ایجاد جدول

```
CREATE TABLE TestTable
    (id BIGINT,
    fname String,
    lname String)
    STORED AS PARQUET;
```

کارعملی

- ایجاد پارتیشن در جدول

```
1 CREATE TABLE TestTable
2     (id BIGINT,
3     fname String,
4     lname String) PARTITIONED BY(m_date_id INT) STORED AS PARQUET;
5
```

کارعملی

- ایجاد پارتیشن و باکت در جدول

```
set hive.enforce.bucketing=true;

create table testTable1
    (id int,
    fname string,
    lname string,
    date_id INT) partitioned by (month_id int)
    clustered by (date_id) into 4 buckets stored as parquet;
```

کارعملی Hive , Impala

کار عملی

- نمایش مسیر داده‌ها
- اجرای کوئری در hive
- اجرای کوئری در impala
- مقایسه سرعت hive و impala

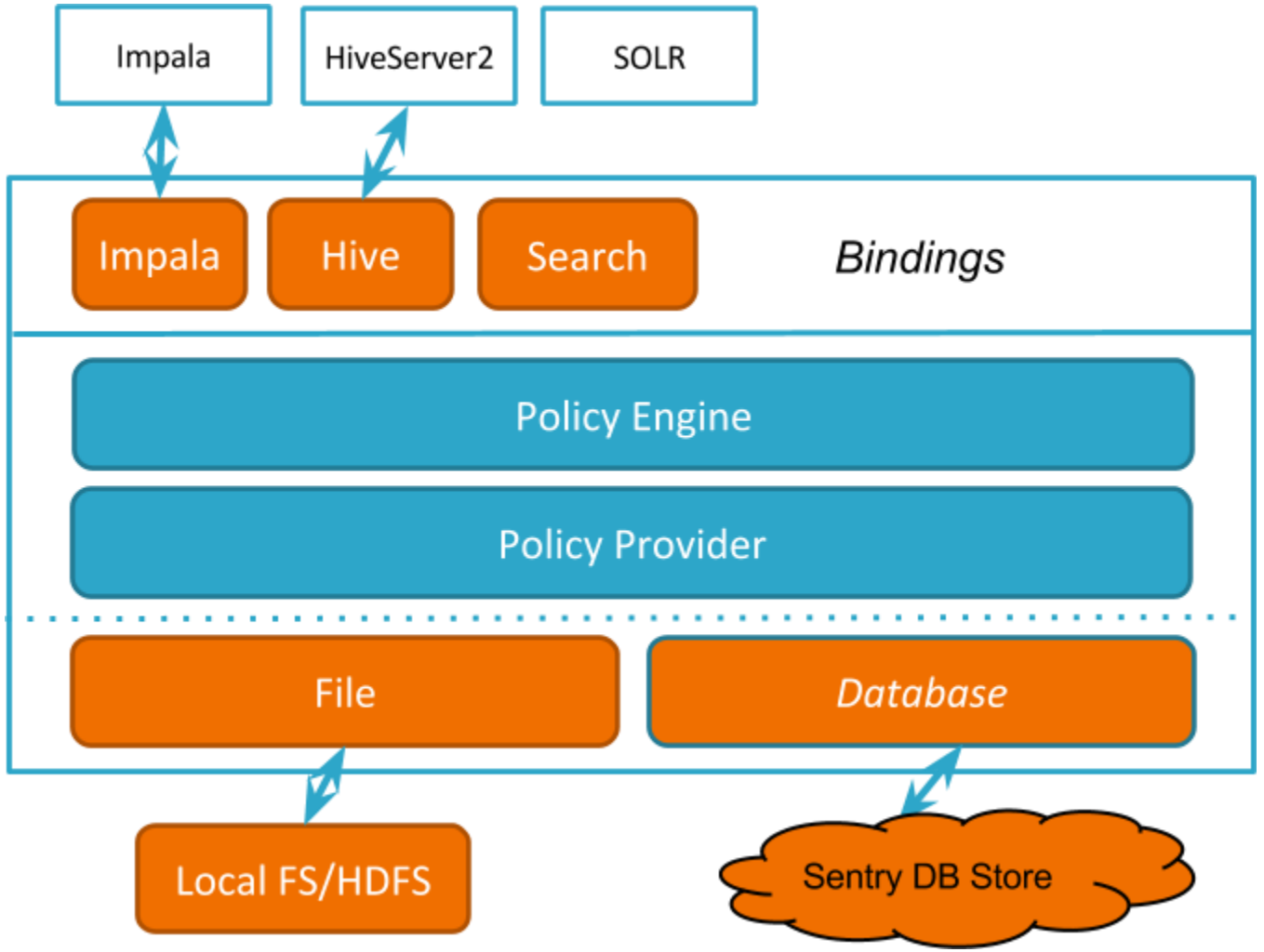
Apache Sentry

Sentry چیست؟

- Role-based Authorization module for Hadoop
- قابلیت ایجاد Role و سطح دسترسی به کاربران
- تعریف دسترسی برای Hive, Impala, HDFS, Solr

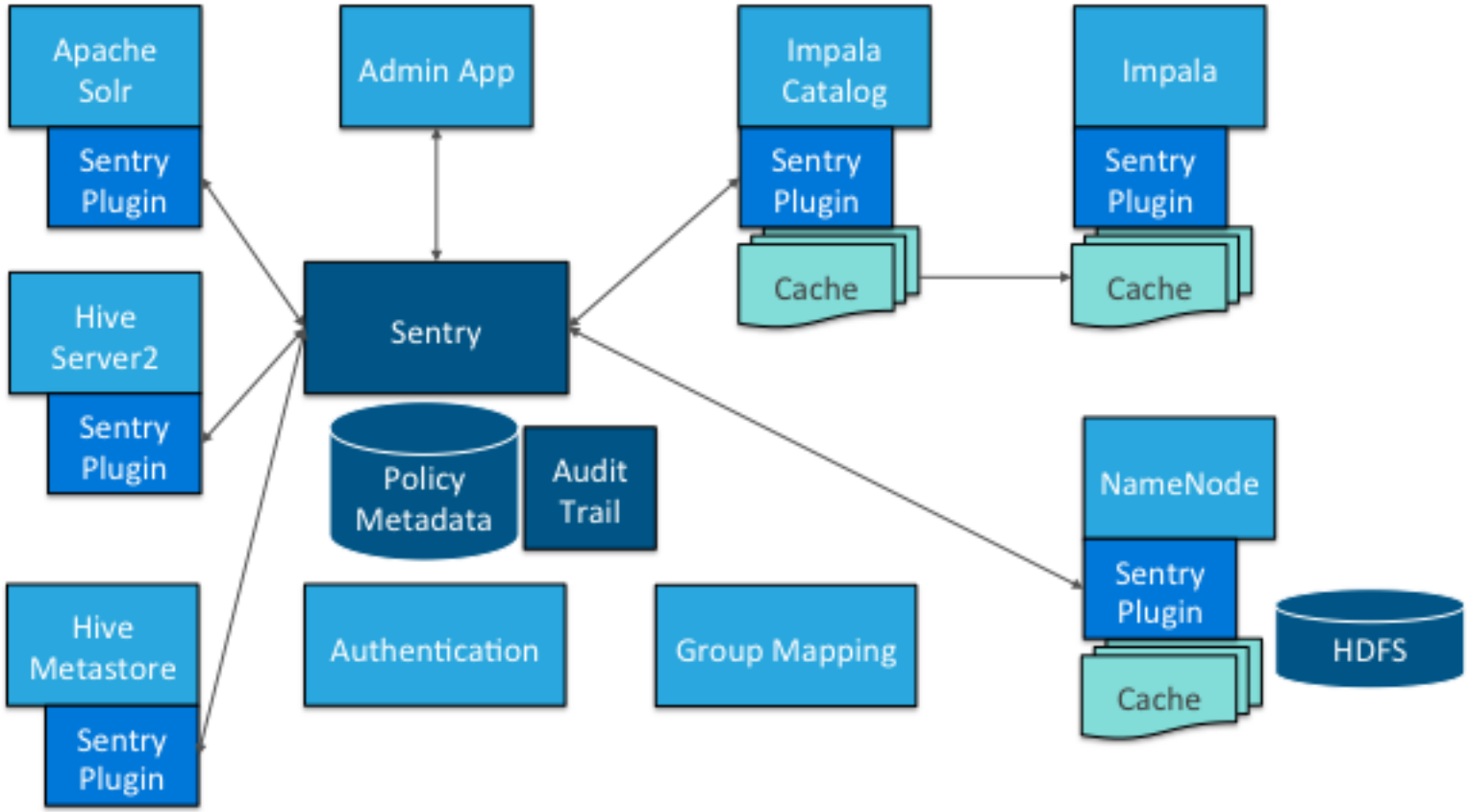
Apache Sentry

معماری Sentry



Apache Sentry

معماری Sentry



کار عملی با Sentry

```
1 SHOW ROLES;
2 SHOW GRANT ROLE admin_role;
3
4 GRANT SELECT
5
```

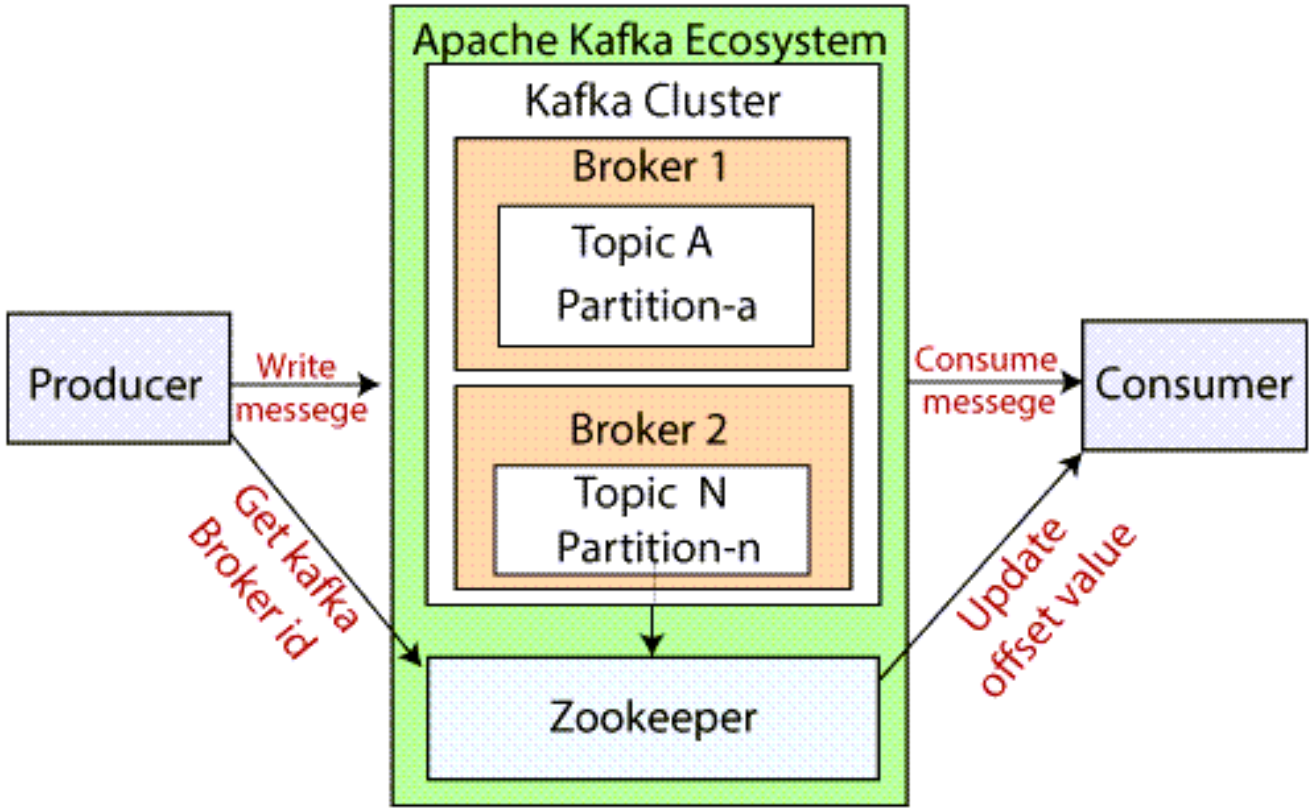
- ON DATABASE keyword
- ON SERVER keyword
- ON TABLE keyword
- ON URI keyword

Apache Kafka

Kafka چیست؟

- یک چارچوب نرم‌افزاری توزیع شده که برای streaming داده‌ها استفاده می‌شود.
- قابلیت پشتیبانی چند مصرف کننده
- قابلیت پشتیبانی پارتیشن

معماری Kafka



معماری Kafka

- Zookeeper
- Broker
- Topic
- Partition
- Producer و Consumer
- Offset
- Consumer group
- retention

معماری Kafka

وظیفه Zookeeper

- **Controller Election**

  تعیین leader/follower برای پارتیشن ها

  تعیین leader جدید درصورت از دست دادن یک نود

- **Cluster Membership**

  تشخیص نودهای از دست داده و نودهای جدید اضافه شده

- **Topic Configuration**

  نگهداری لیست topicهای موجود

  تعداد پارتیشن هر topic

  مکان هر replicaی topic

وظیفه Zookeeper

- **Access Control Lists (ACLs)**

  دسترسی خواندن و نوشتن روی هر topic

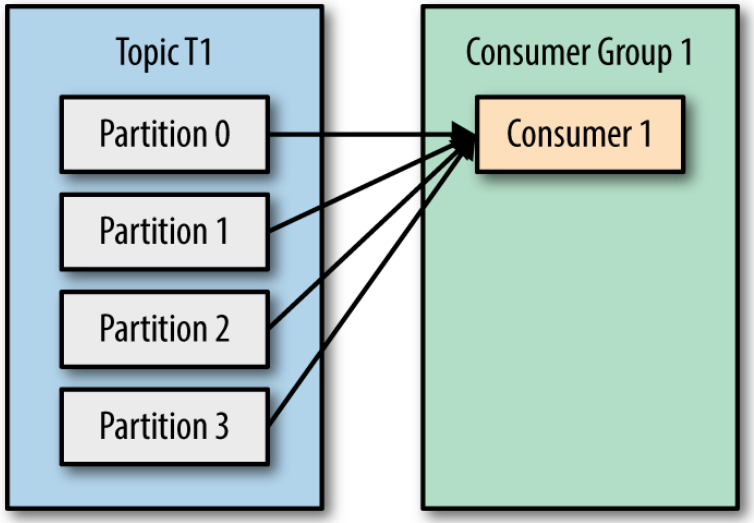
  لیست Consumer Groupهای مربوط به هر topic

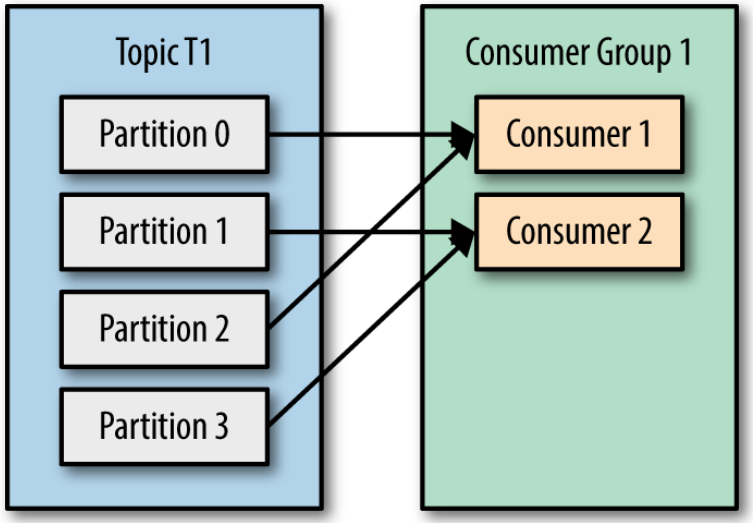
  نگهداری offset مربوط به هرConsumer Group روی هر Partition

- **Quotas**

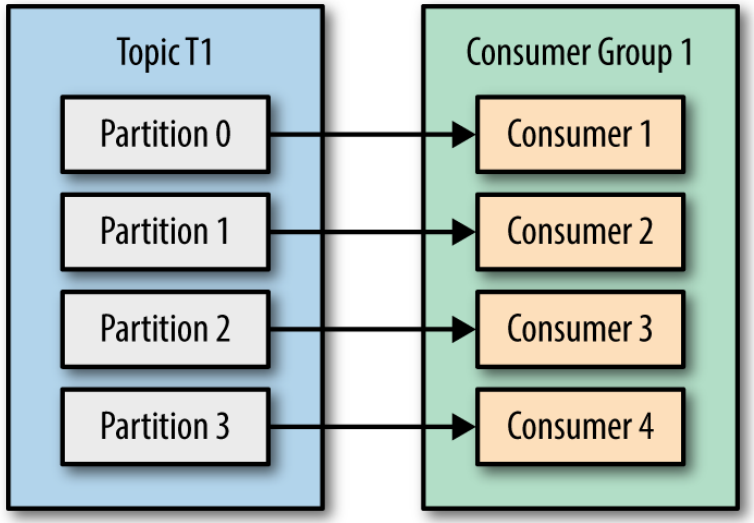
  تعیین read/write به ازای هر کلاینت

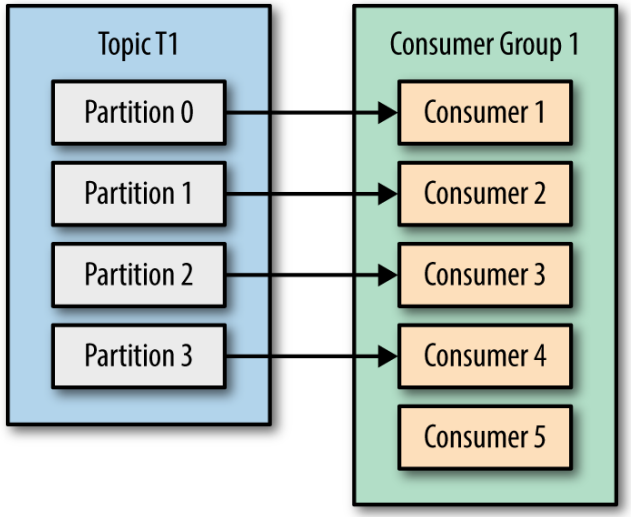
Kafka Consumer



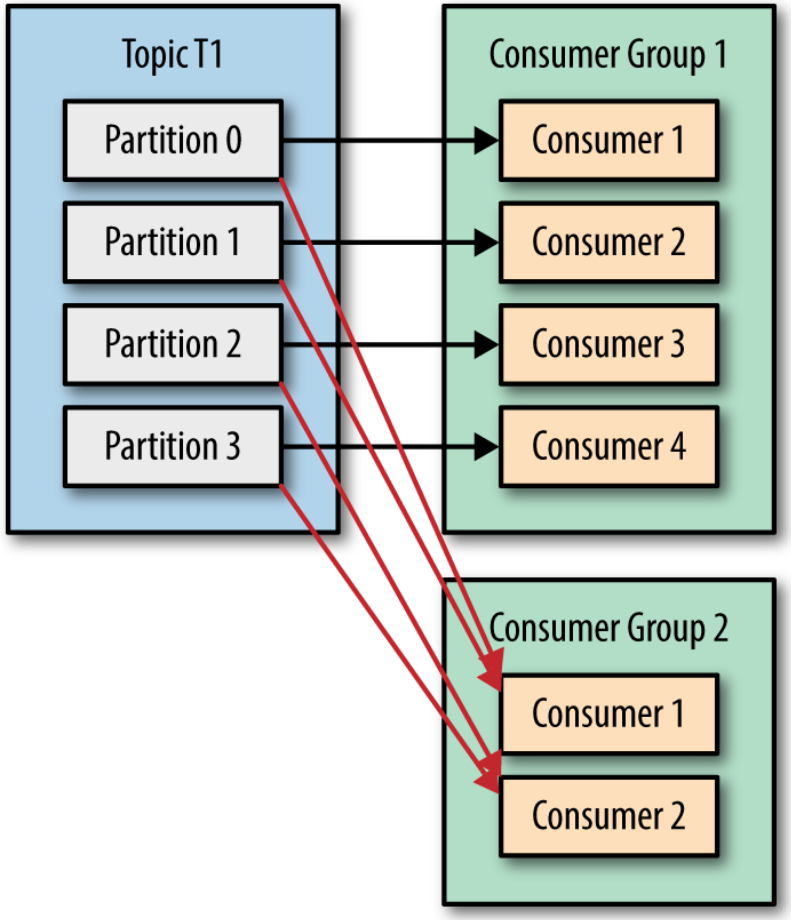


Kafka Consumer



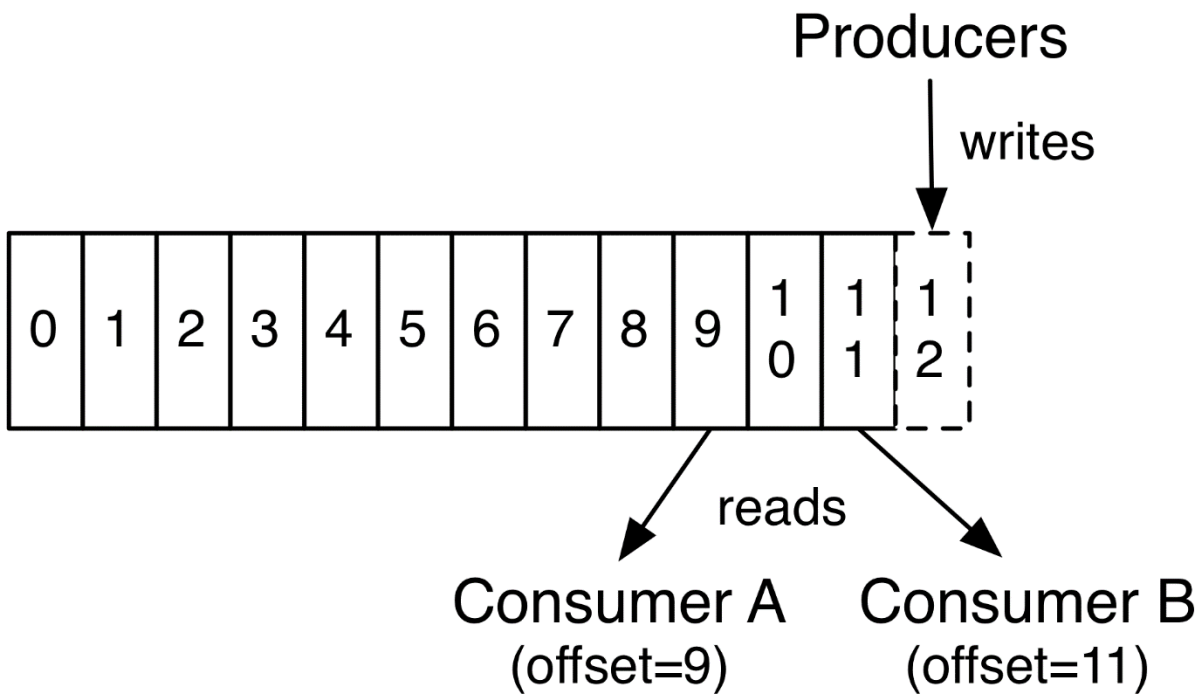


Kafka Consumer



Consumer Group

خوندن همزمان یک Partition از یک Topic با دو Consumer Group متفاوت و نوشتن دیتای جدید با Producer



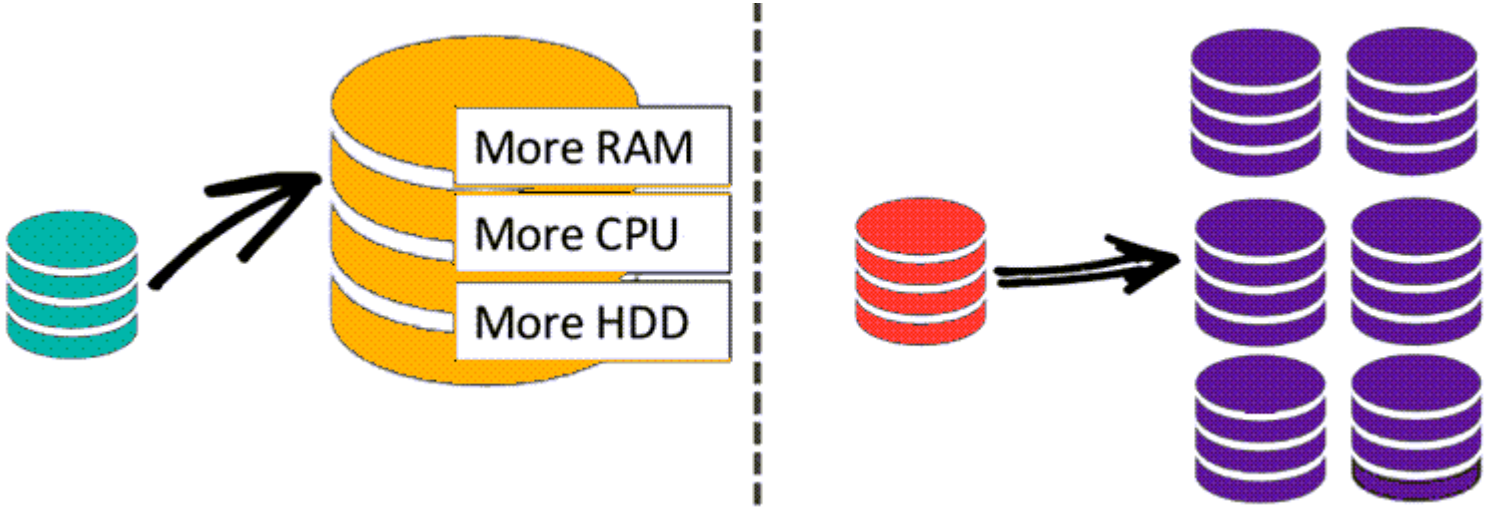
کارعملی Kafka

- نمایش کانفیگ کلاستر Kafka
- اجرای دستورات در console

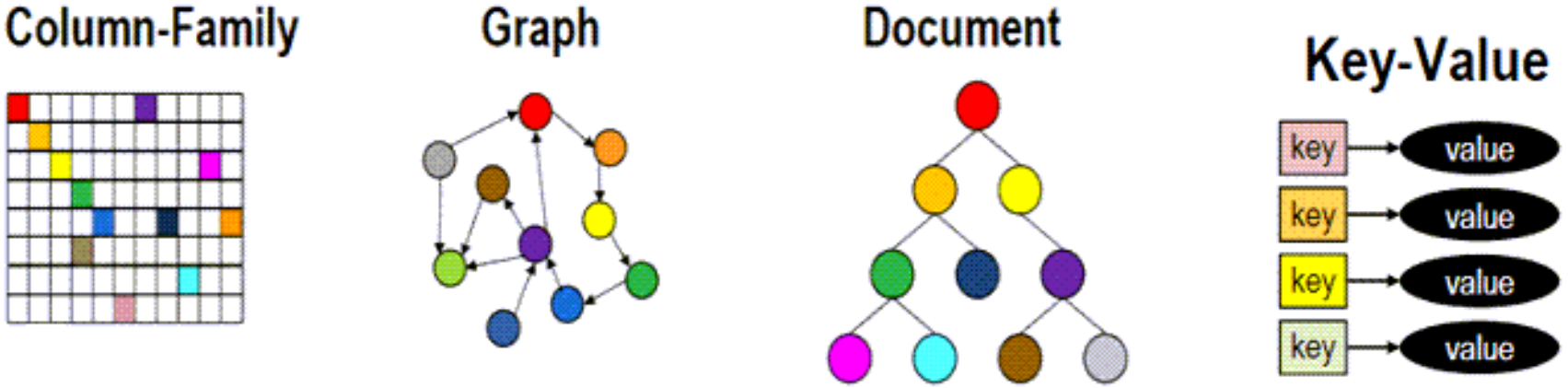
```
# The send buffer (SO_SNDBUF) used by the socket server
socket.send.buffer.bytes=102400

# The receive buffer (SO_RCVBUF) used by the socket server
socket.receive.buffer.bytes=102400

# The maximum size of a request that the socket server will accept (protection against OOM)
socket.request.max.bytes=104857600


############################# Log Basics #############################

# A comma seperated list of directories under which to store log files
log.dirs=/var/kafka/data

# The default number of log partitions per topic. More partitions allow greater
# parallelism for consumption, but this will also result in more files across
# the brokers.
num.partitions=2
```

NoSQL DB چیست؟

- یک سیستم مدیریت داده غیر رابطه‌ای
- ساختار ثابت ندارد
- مقیاس پذیری آسان
- ذخیره سازی توزیع شده



انواع NoSQL DB



انواع NoSQL DB

Col1	Col2	Col3	Col4
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data

Document 1

```
{
  "prop1": data,
  "prop2": data,
  "prop3": data,
  "prop4": data
}
```

Document 2

```
{
  "prop1": data,
  "prop2": data,
  "prop3": data,
  "prop4": data
}
```

Document 3

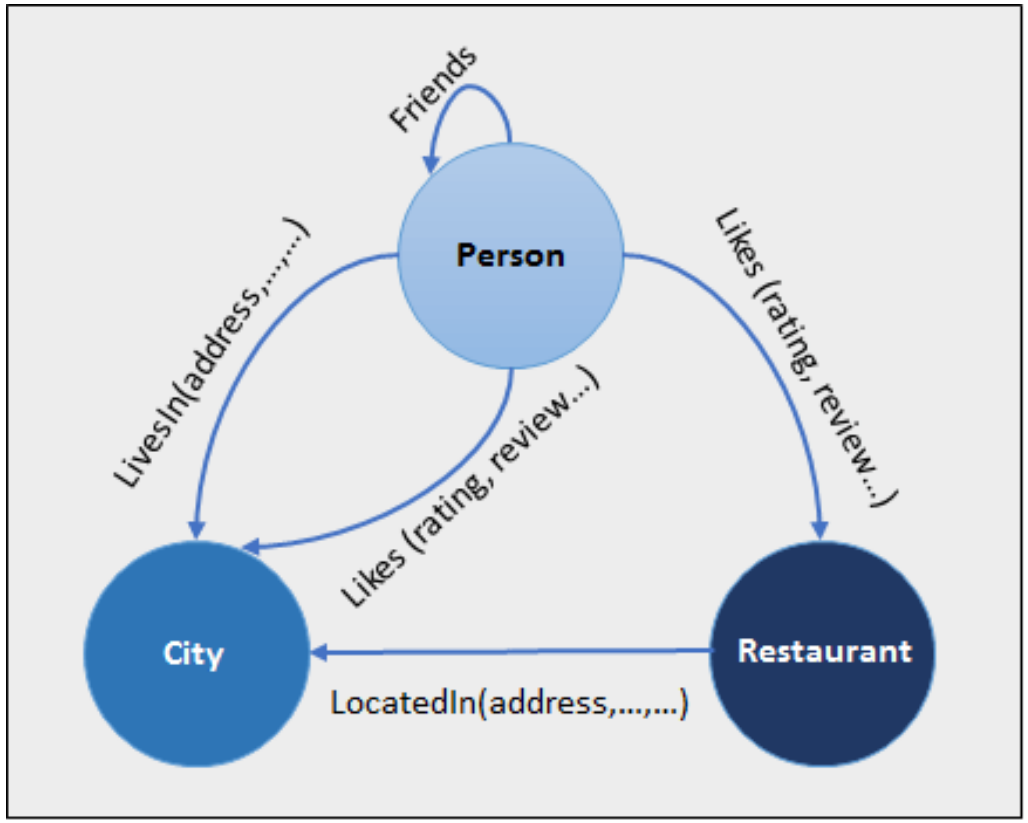
```
{
  "prop1": data,
  "prop2": data,
  "prop3": data,
  "prop4": data
}
```

انواع NoSQL DB

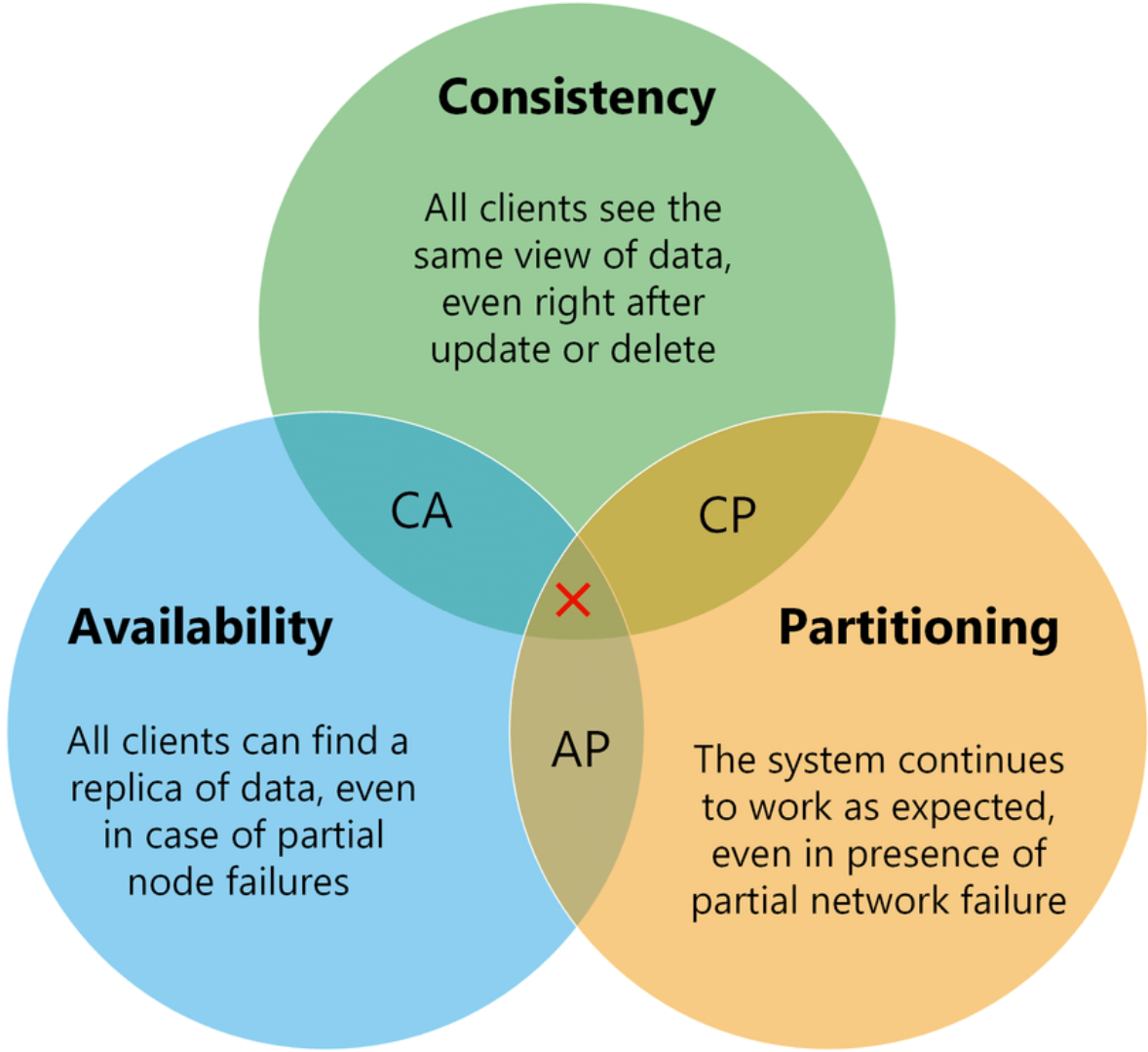
ColumnFamily			
Row Key	Column Name		
	Key	Key	Key
	Value	Value	Value
	Column Name		
	Key	Key	Key
	Value	Value	Value

Key	Value
Name	Joe Bloggs
Age	42
Occupation	Stunt Double
Height	175cm
Weight	77kg

انواع NoSQL DB

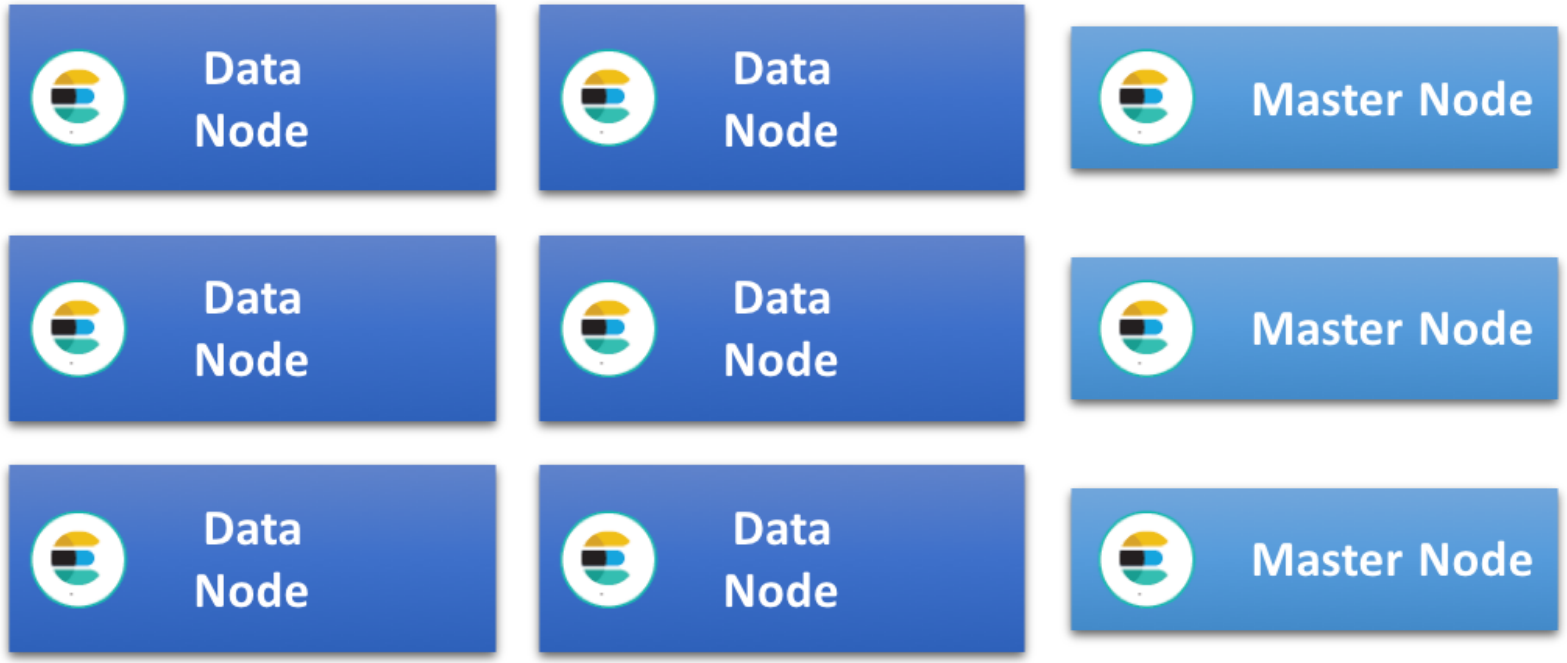


تئوری CAP


Consistency
All clients see the same view of data, even right after update or delete
CA
CP
Availability
All clients can find a replica of data, even in case of partial node failures
AP
Partitioning
The system continues to work as expected, even in presence of partial network failure

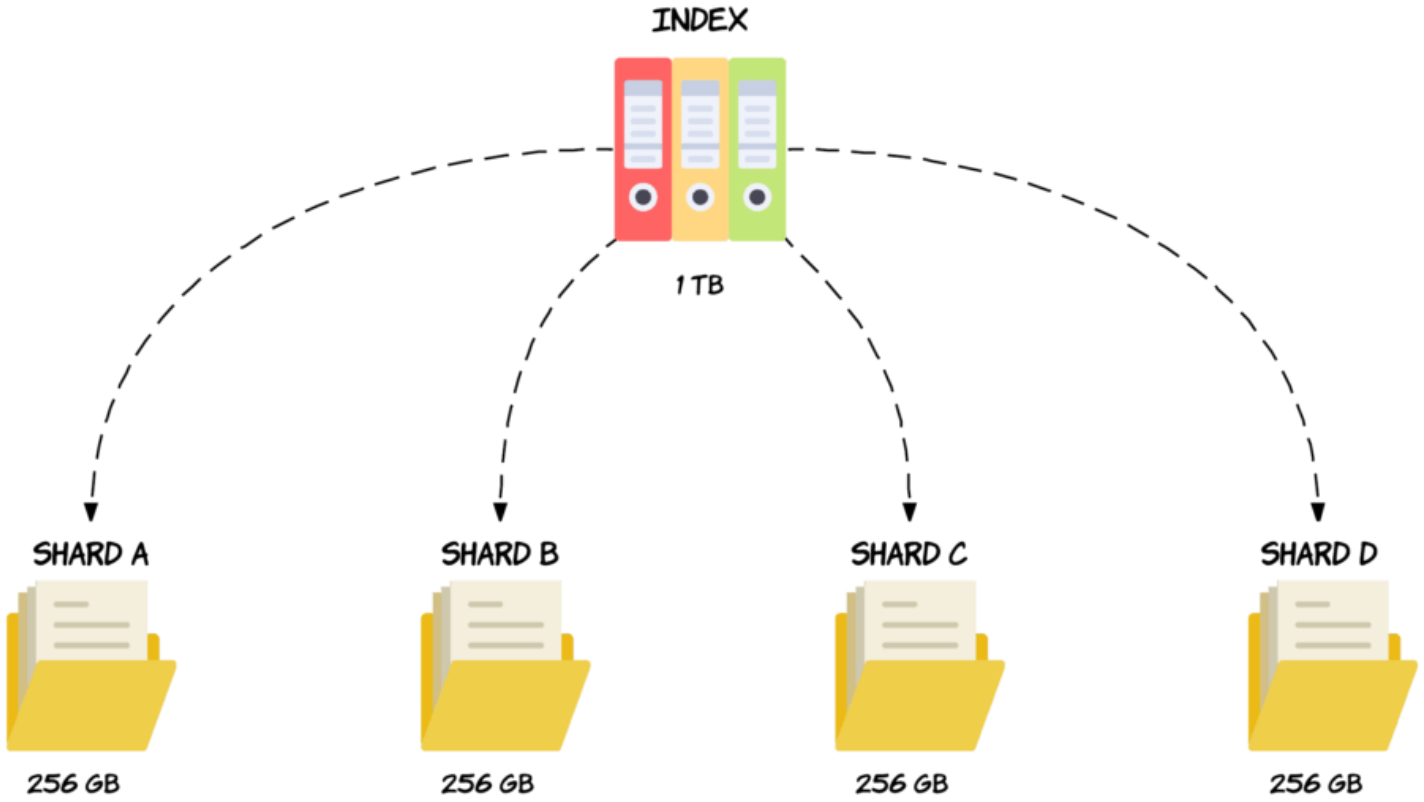
معماری Elasticsearch

- Master Node
- Data Node
- Ingest Node

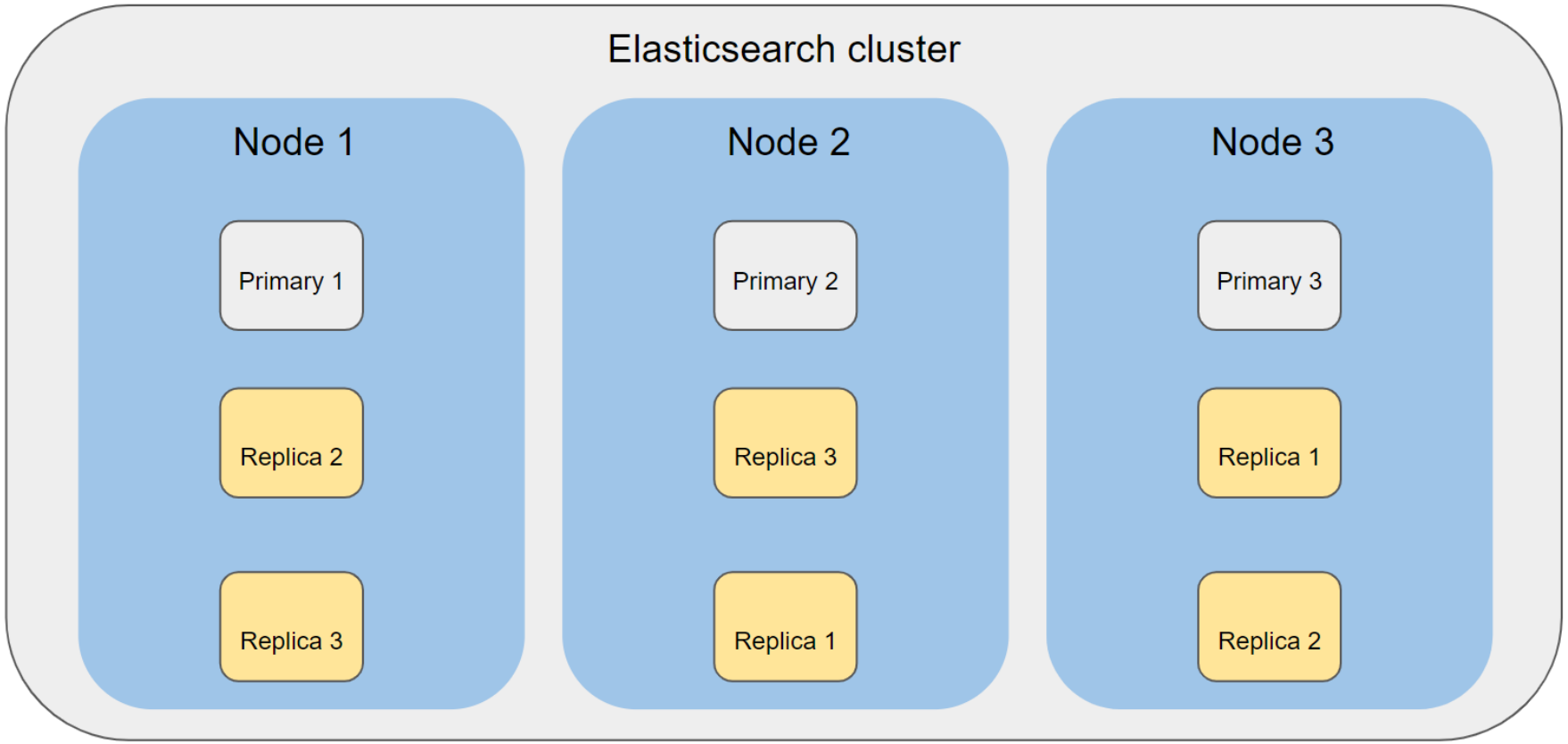


معماری Elasticsearch

- Document
- Index
- Shard
- Replica



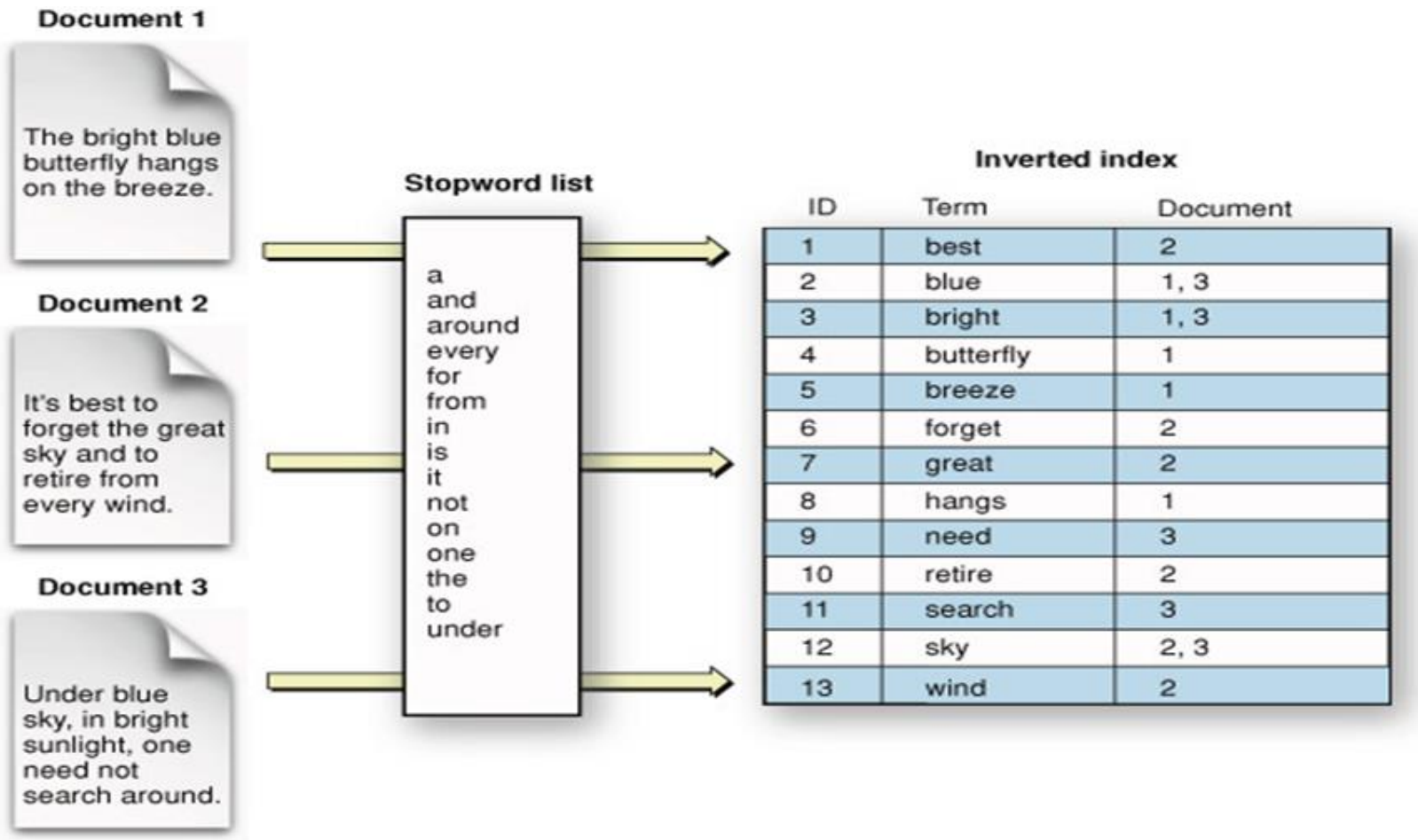
معماری Elasticsearch



معماری Elasticsearch

<table>
<tr><td colspan="8">Elasticsearch Index</td></tr>
<tr><td colspan="2">Elasticsearch shard</td><td colspan="2">Elasticsearch shard</td><td colspan="2">Elasticsearch shard</td><td colspan="2">Elasticsearch shard</td></tr>
<tr><td colspan="2">Lucene index</td><td colspan="2">Lucene index</td><td colspan="2">Lucene index</td><td colspan="2">Lucene index</td></tr>
<tr><td>Segment</td><td>Segment</td><td>Segment</td><td>Segment</td><td>Segment</td><td>Segment</td><td>Segment</td><td>Segment</td></tr>
</table>

معماری Elasticsearch



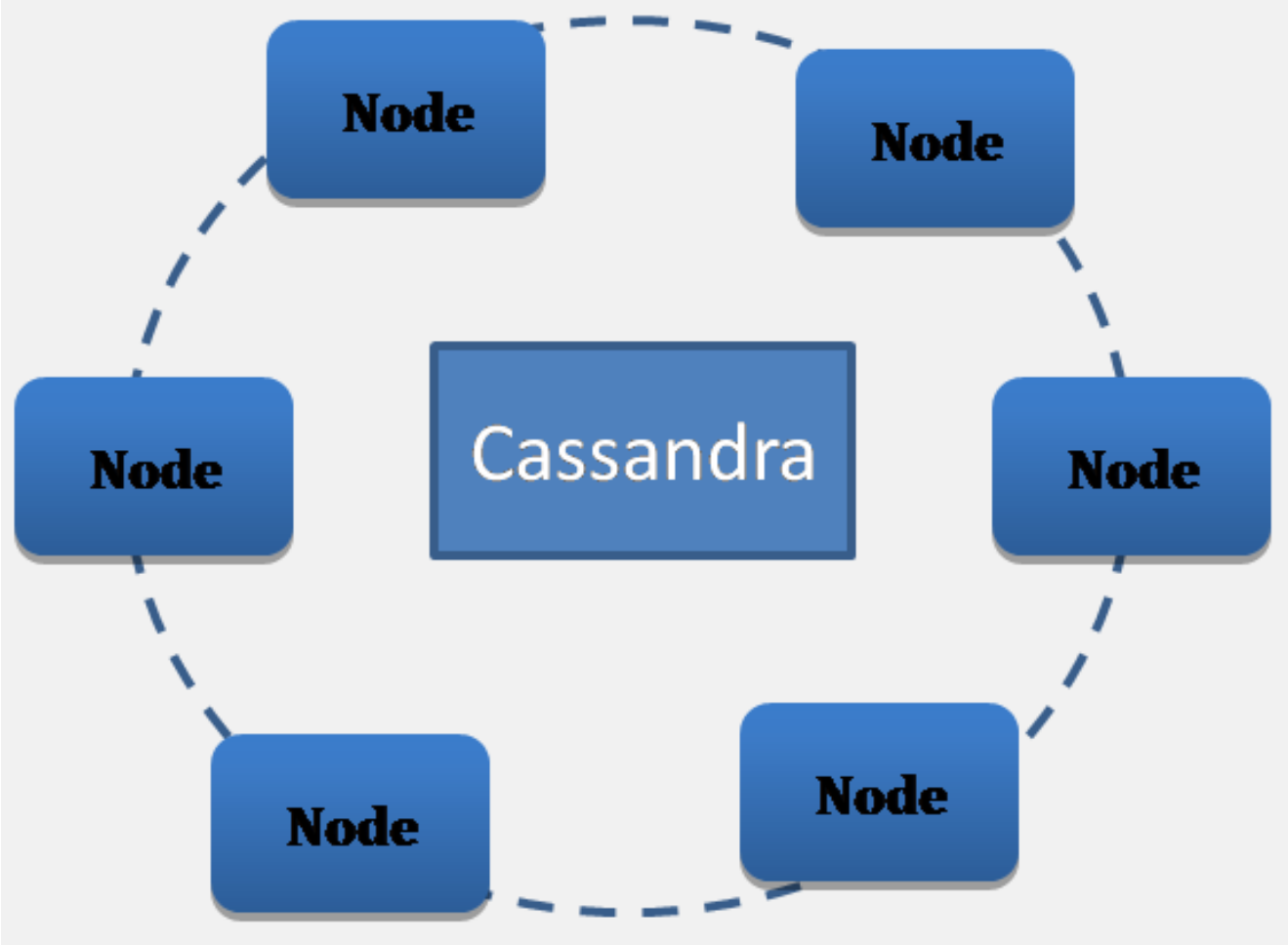
کارعملی روی Elasticsearch

- ساختن ایندکس
- ساختار ایندکس (Mapping)
- تنظیمات ایندکس
- جستجو روی ایندکس
- استفاده از Kibana

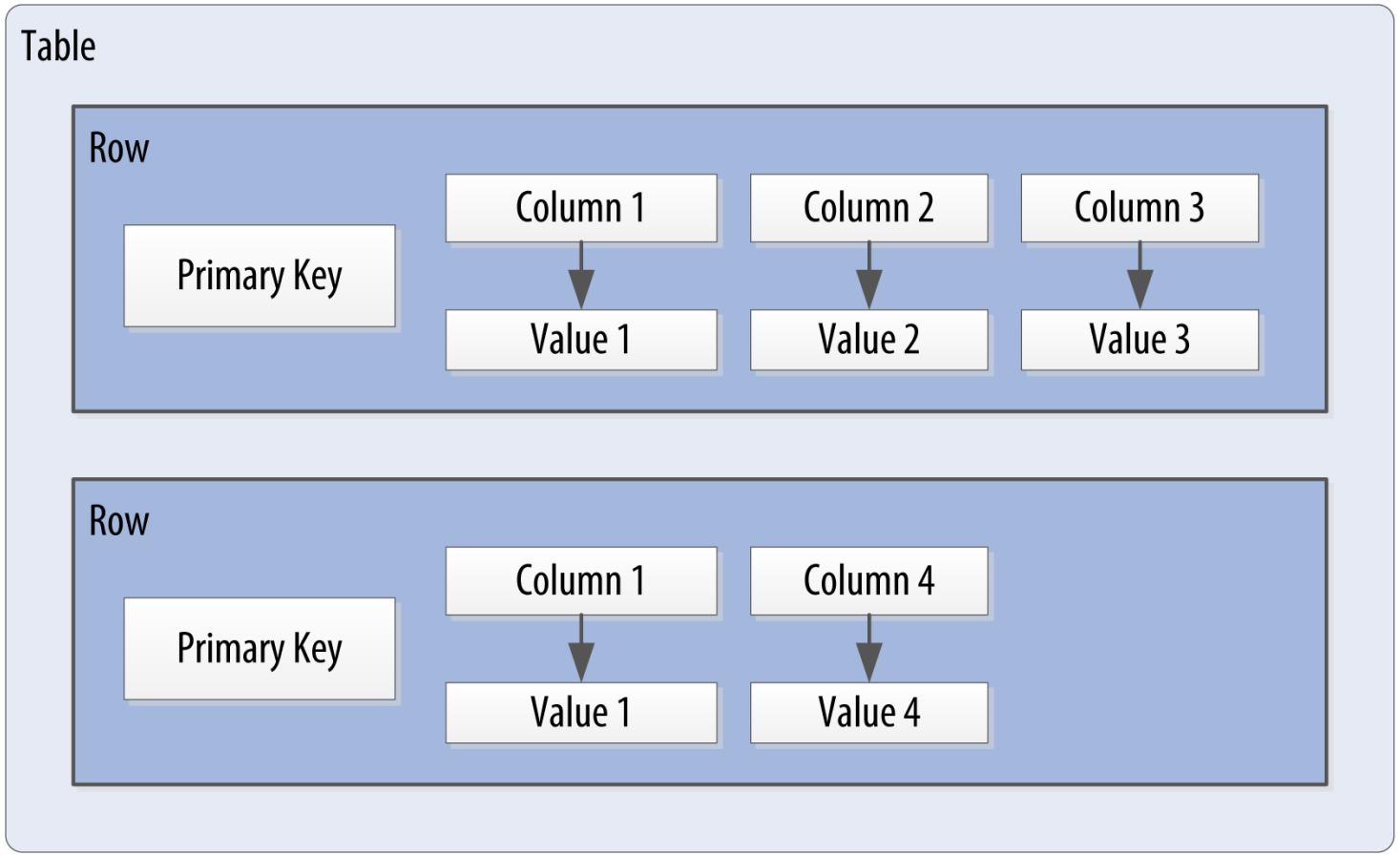
```
GET _cat/indices?v
GET doc_test/_mapping
```

```
{
  "doc_test" : {
    "mappings" : {
      "properties" : {
        "clientMessageId" : {
          "type" : "text",
          "fields" : {
            "keyword" : {
              "type" : "keyword",
              "ignore_above" : 256
            }
          }
        },
        "duration" : {
          "type" : "long"
        },
        "errorCode" : {
          "type" : "long"
        },
```

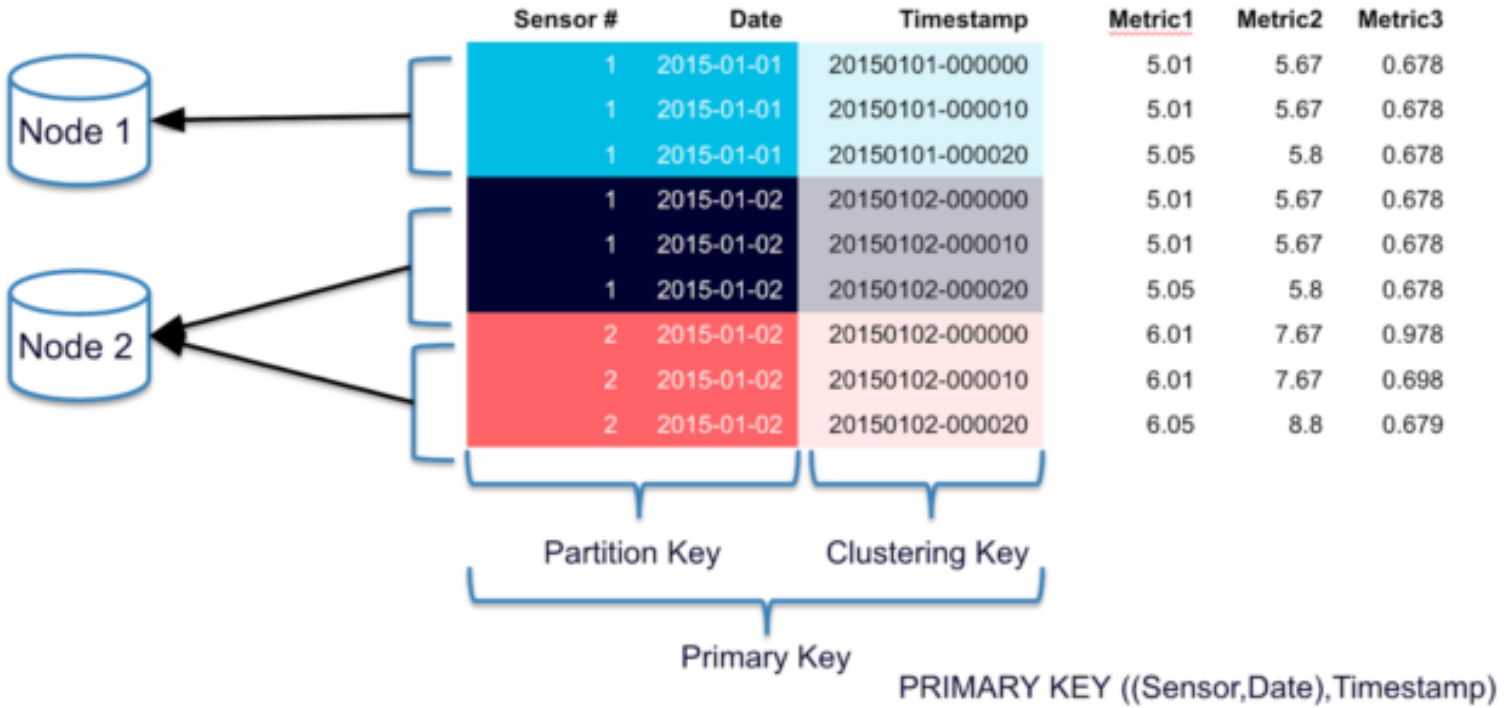
معماری Cassandra



ساختار داده در Cassandra



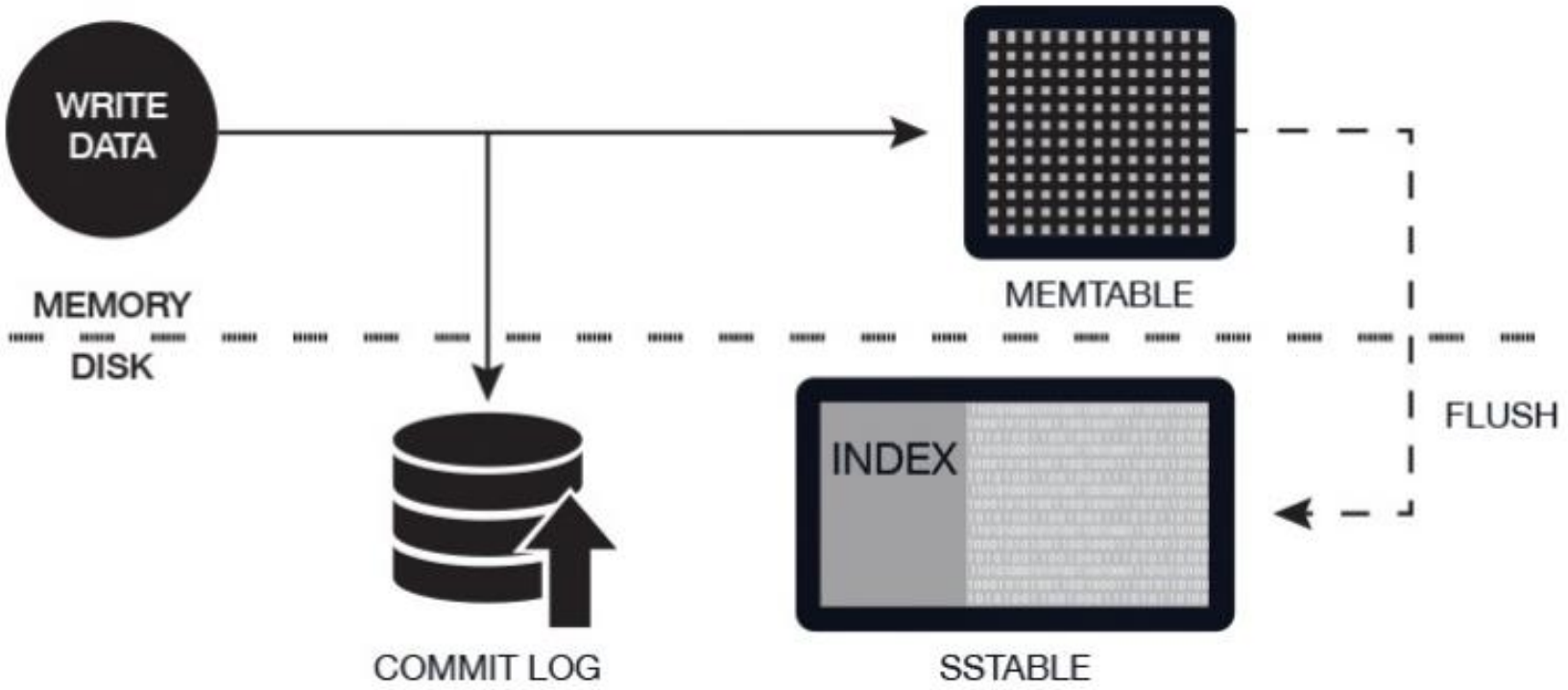
پارتیشن در Cassandra



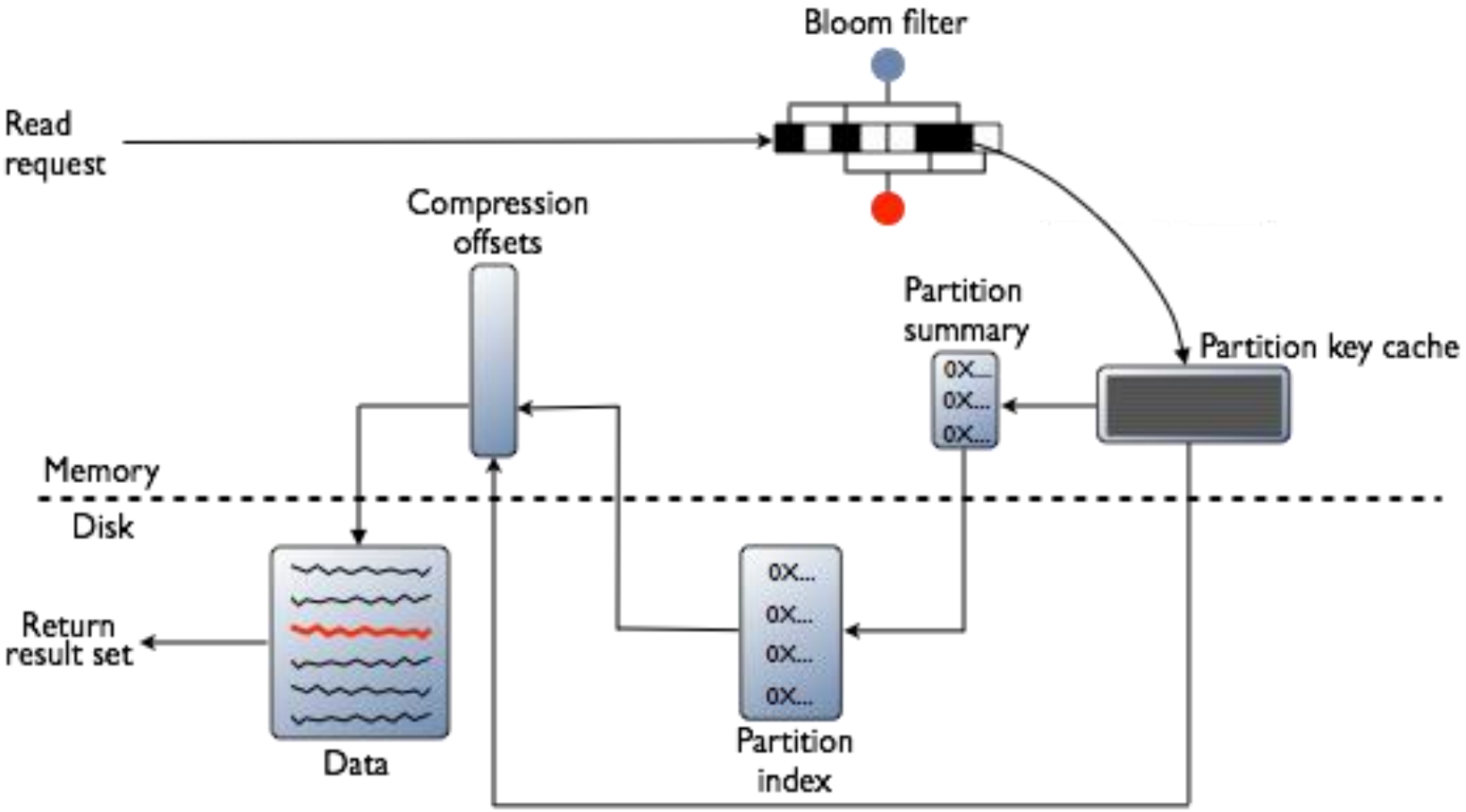
Sensor #	Date	Timestamp	Metric1	Metric2	Metric3
1	2015-01-01	20150101-000000	5.01	5.67	0.678
1	2015-01-01	20150101-000010	5.01	5.67	0.678
1	2015-01-01	20150101-000020	5.05	5.8	0.678
1	2015-01-02	20150102-000000	5.01	5.67	0.678
1	2015-01-02	20150102-000010	5.01	5.67	0.678
1	2015-01-02	20150102-000020	5.05	5.8	0.678
2	2015-01-02	20150102-000000	6.01	7.67	0.978
2	2015-01-02	20150102-000010	6.01	7.67	0.698
2	2015-01-02	20150102-000020	6.05	8.8	0.679

PRIMARY KEY ((Sensor,Date),Timestamp)

نوشتن در Cassandra



خواندن از Cassandra



با تشکر ؟؟؟

فناپ
FANAP

وزارت علوم، تحقیقات و فناوری
دانشگاه خاتم
Khatam University