COPENHAGEN
INTERNATIONAL
SOFTWARE DEVELOPMENT
CONFERENCE 2014

goto; conference

COPENHAGEN
INTERNATIONAL
SOFTWARE DEVELOPMENT
CONFERENCE

Data at the Speed of your Users

Apache Cassandra and Spark for simple, distributed, near real-time stream processing.

GOTO Copenhagen 2014

Rustam Aliyev

Solution Architect at 

@rstml

Big Data?

Photo: Flickr / Watches En Masse

Volume

Variety

Velocity

Velocity = Near Real Time

Near Real Time?

0.5 sec ≤ Near Real Time ≤ 60 sec

Use Cases

Photo: Flickr / Swiss Army / Jim Pennucci

Web Analytics

Dynamic Pricing

Recommendation

Fraud Detection

Architecture

Photo: Ilkin Kangarli / Baku Haydar Aliyev Center

Architecture Goals

Low Latency

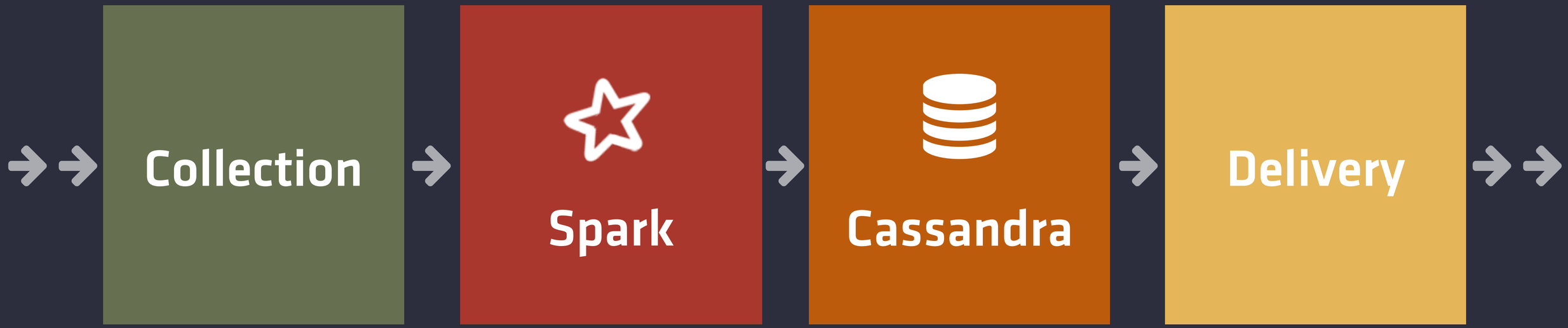
High Availability

Horizontal Scalability

Simplicity

Stream Processing

Stream Processing



Cassandra

Distributed Database

Photo: Flickr / Hypostyle Hall / Jorge Láscar

Data Model

Partition

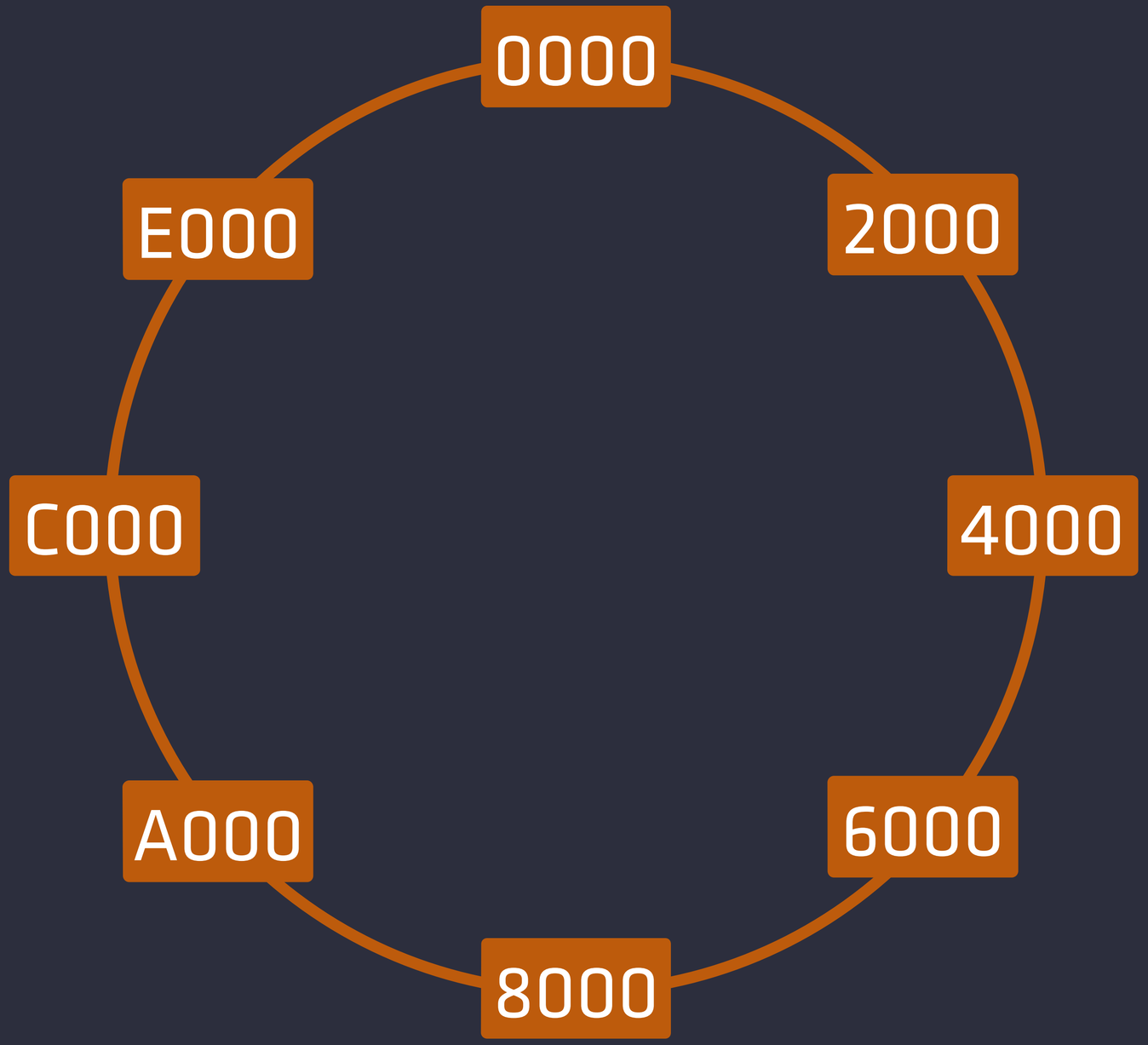
Partition

Table

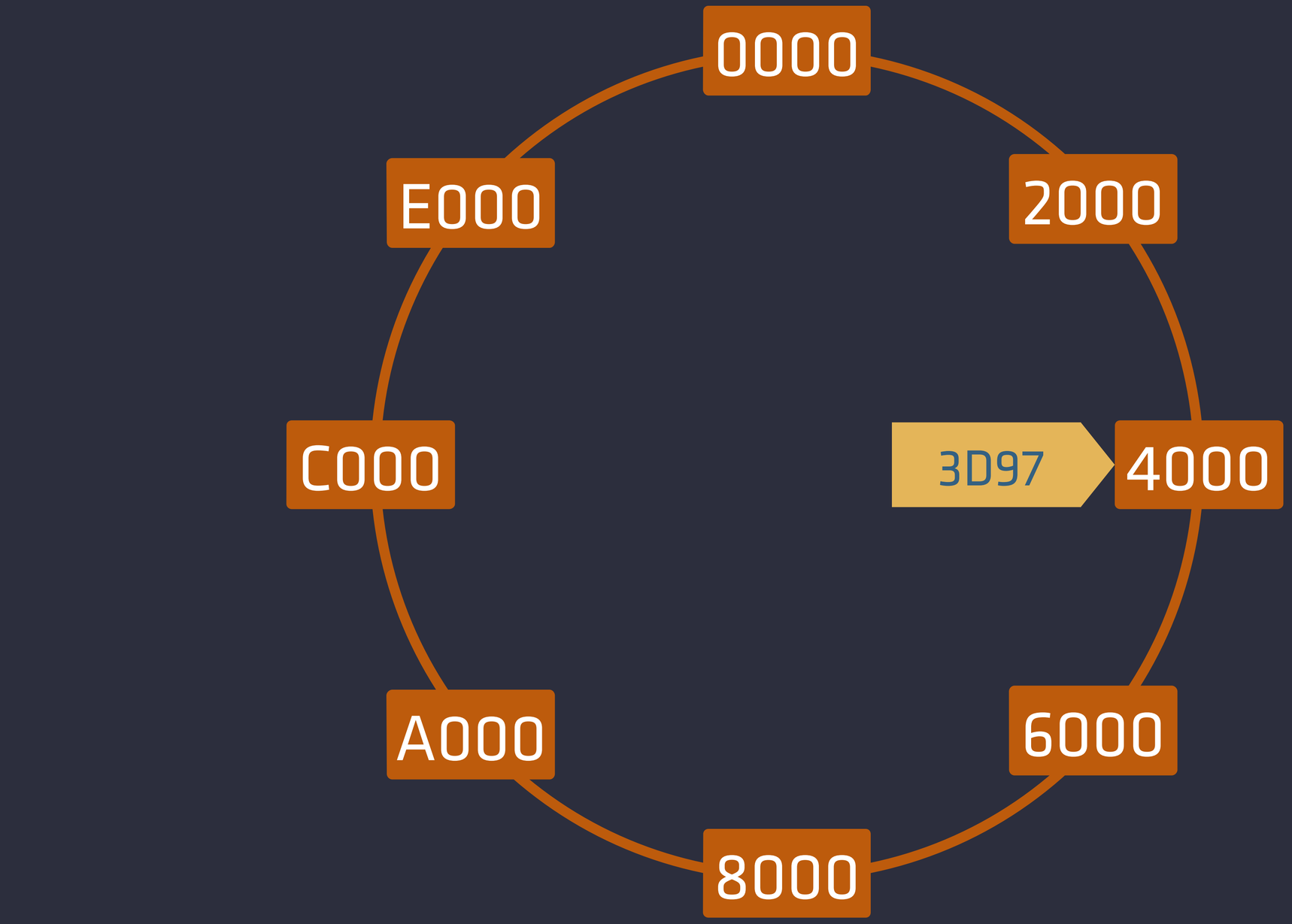
Nexus 5	os: *Android*	storage: *32GB*	version: *4.4*	weight: *130g*
iPhone 6	os: *iOS*	storage: *64GB*	version: *8.0*	weight: *129g*
	os:	memory:	version:	weight:

Distribution

Nexus 5



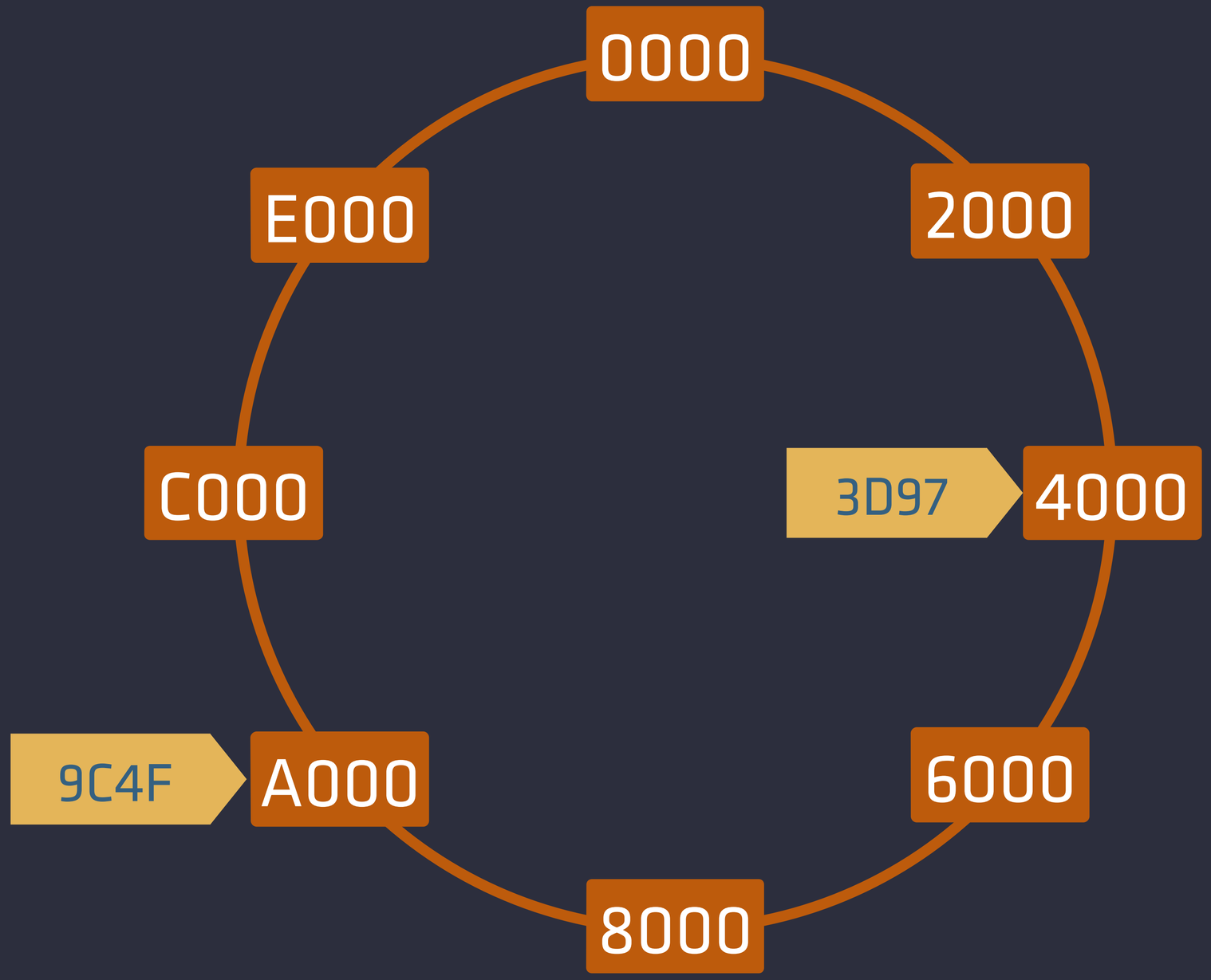
iPhone 6



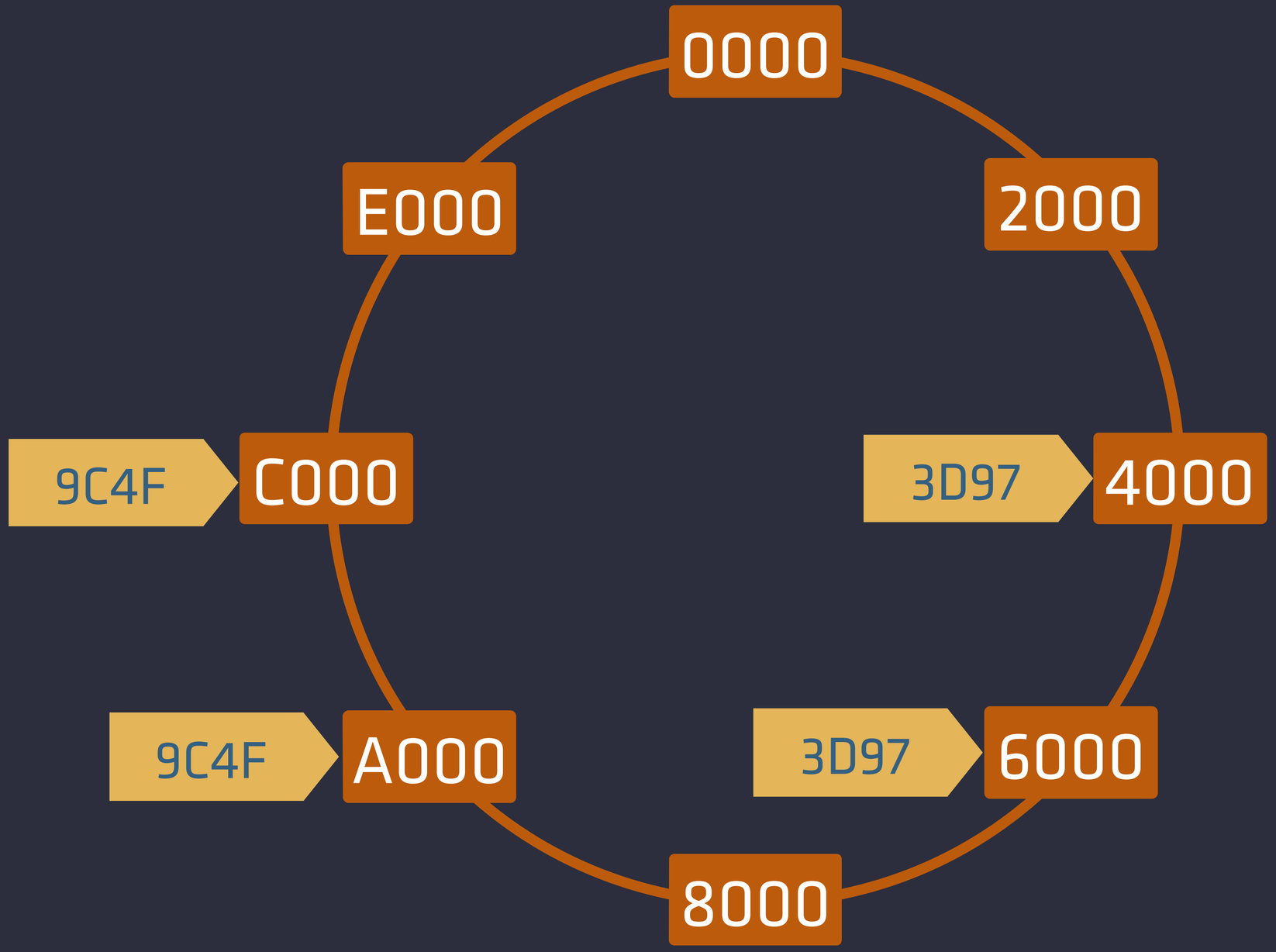
DATASTAX

Replication

1 replica



2 replicas



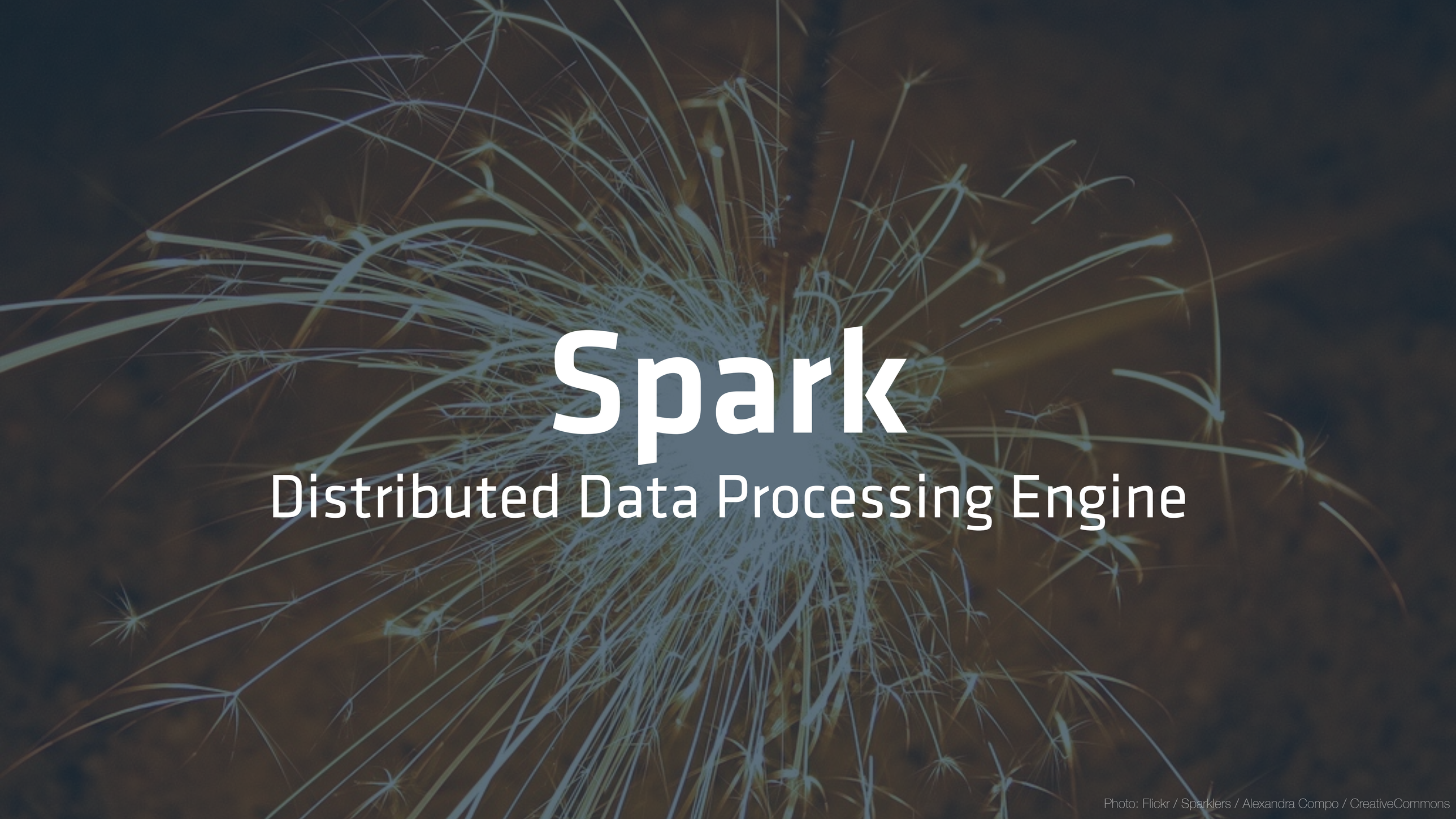
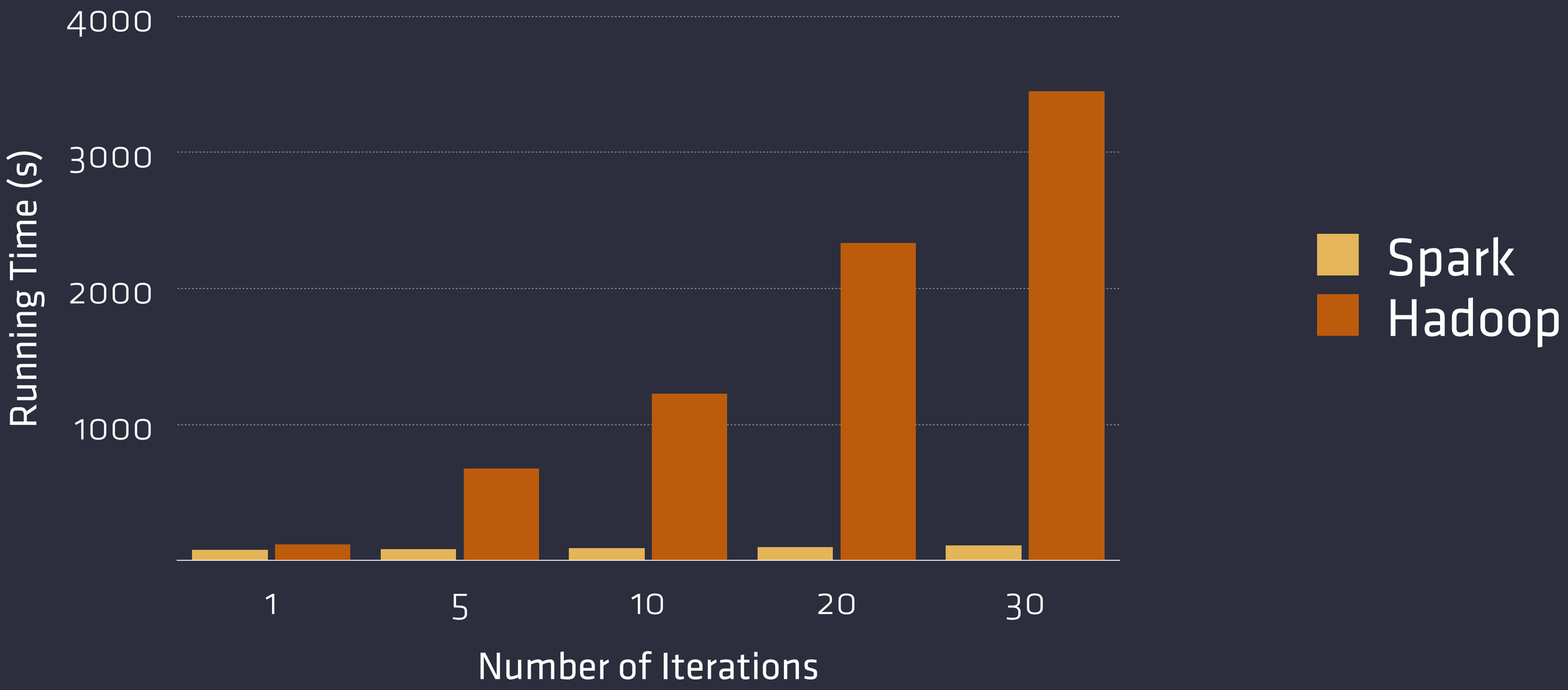


Photo: Flickr / Sparklers / Alexandra Compo / CreativeCommons

Fast

In-memory

Logistic Regression



Easy

map

reduce

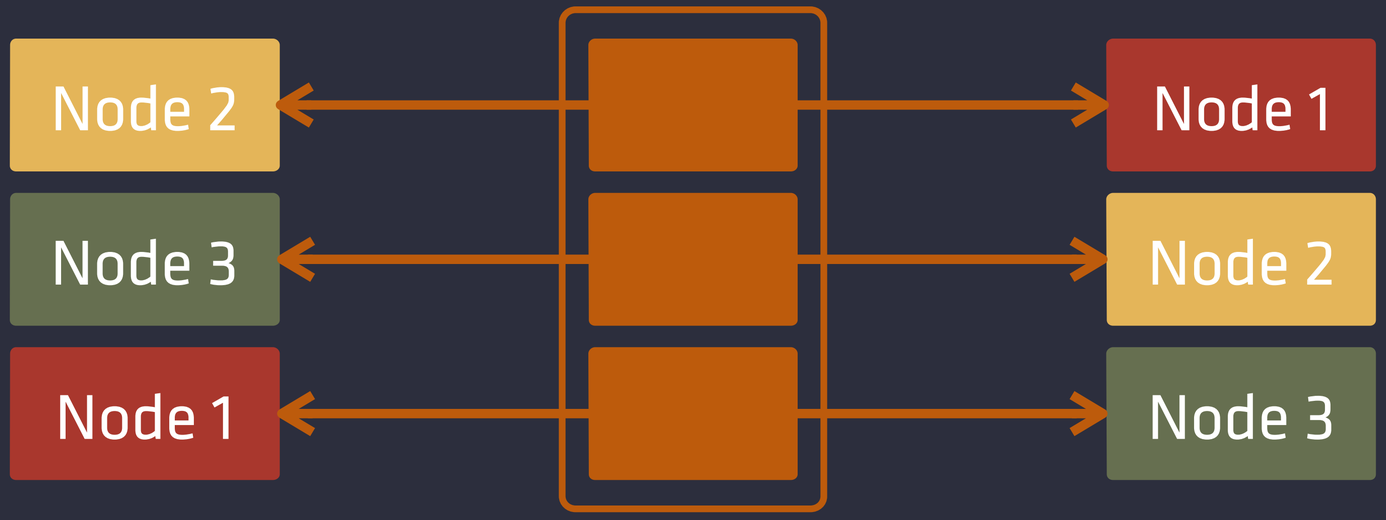
map
filter
groupBy
sort
union
join
leftOuterJoin
rightOuterJoin

reduce
count
fold
reduceByKey
groupByKey
cogroup
cross
zip

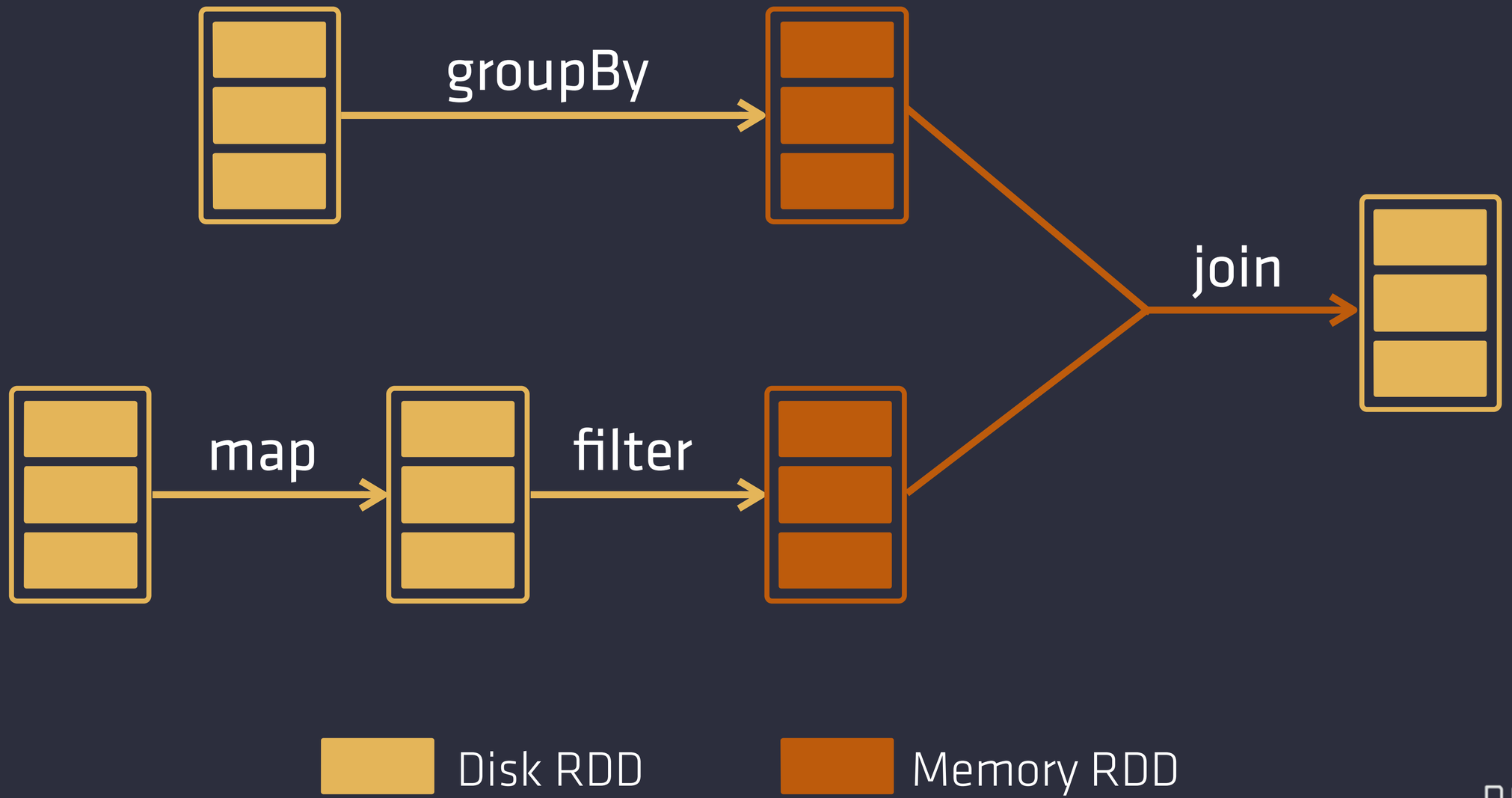
sample
take
first
partitionBy
mapWith
pipe
save
...

RDD

Resilient Distributed Datasets

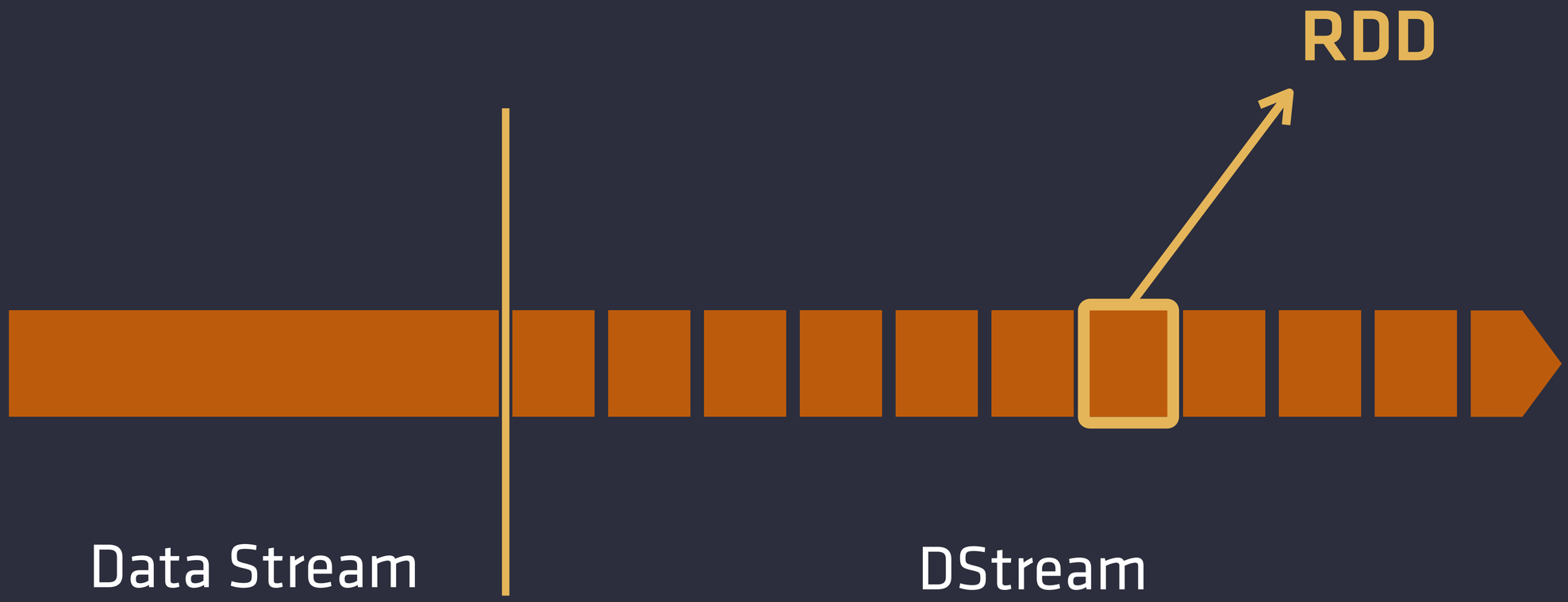


Operator DAG



Spark Streaming

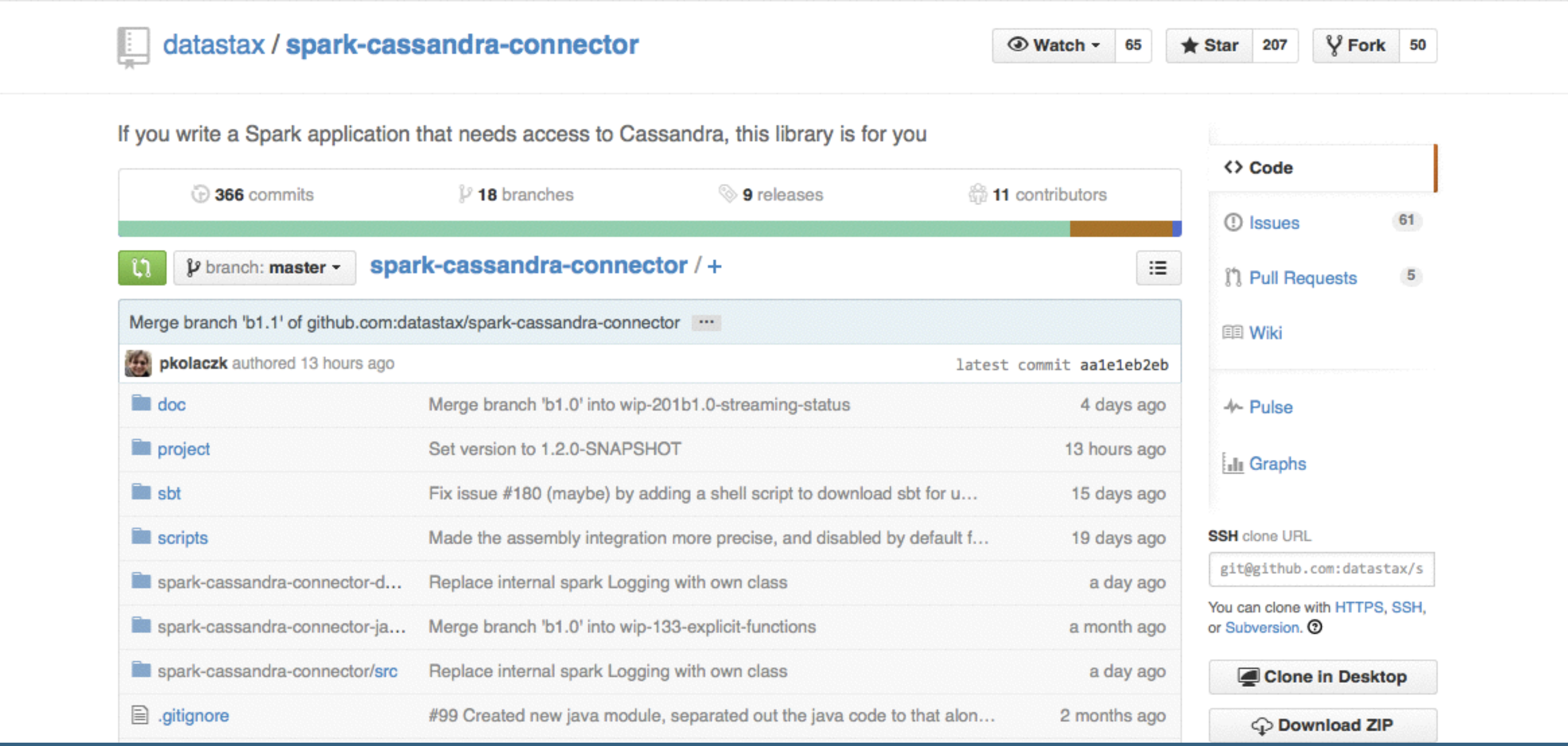
Micro-batching



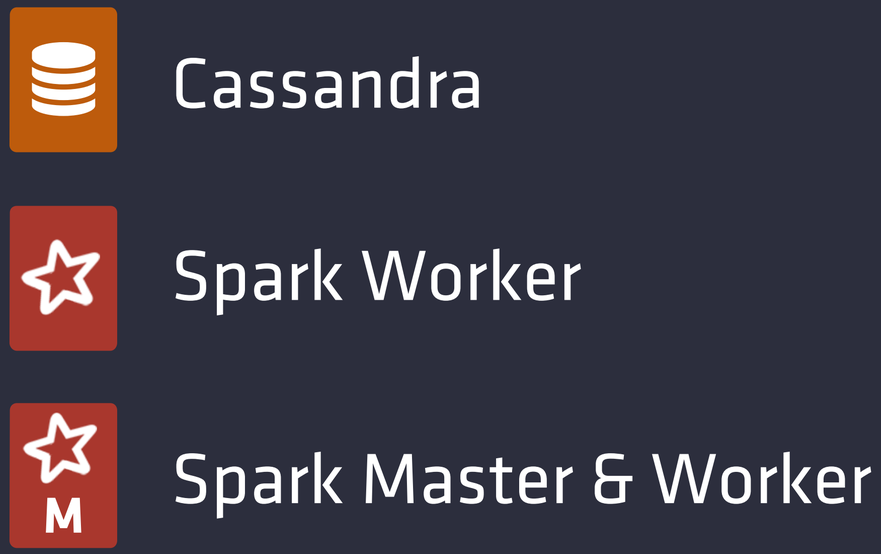
Spark + Cassandra

DataStax Spark Cassandra Connector

DATASTAX



https://github.com/datastax/spark-cassandra-connector

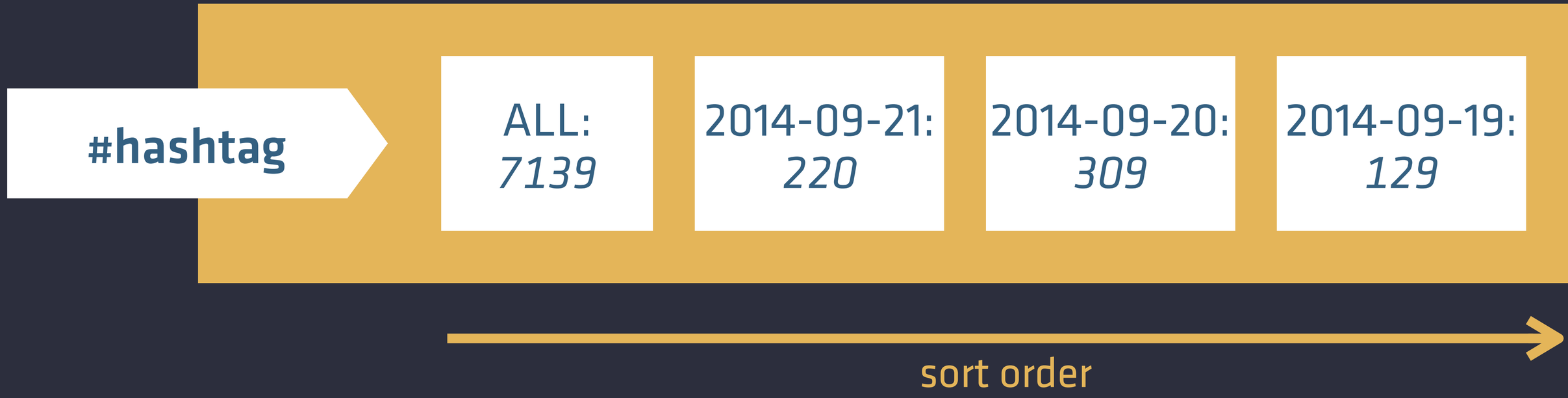

Cassandra
Spark Worker
M
Spark Master & Worker


M
DATASTAX
M

Demo

Twitter Analytics

Cassandra Data Model


#hashtag
ALL:
7139
2014-09-21:
220
2014-09-20:
309
2014-09-19:
129
sort order

```
CREATE TABLE hashtags (
    hashtag text,
    interval text,
    mentions counter,
    PRIMARY KEY((hashtag), interval)
) WITH CLUSTERING ORDER BY (interval DESC);
```

Processing Data Stream

```
import com.datastax.spark.connector.streaming._

val sc = new SparkConf()
  .setMaster("spark://127.0.0.1:7077")
  .setAppName("Twitter-Demo")
  .setJars("demo-assembly-1.0.jar"))
  .set("spark.cassandra.connection.host", "127.0.0.1")

val ssc = new StreamingContext(sc, Seconds(2))

val stream = TwitterUtils.
  createStream(ssc, None, Nil, storageLevel = StorageLevel.MEMORY_ONLY_SER_2)

val hashTags = stream.flatMap(tweet =>
  tweet.getText.toLowerCase.split(" ").
  filter(tags.contains(Seq("#iphone", "#android"))))

val tagCounts = hashTags.map((_, 1)).reduceByKey(_ + _)

val tagCountsAll = tagCounts.map{
  case (tag, mentions) => (tag, mentions, "ALL")
}

tagCountsAll.saveToCassandra(
```

DATASTAX

```
import com.datastax.spark.connector.streaming._

val sc = new SparkConf()
  .setMaster("spark://127.0.0.1:7077")
  .setAppName("Twitter-Demo")
  .setJars("demo-assembly-1.0.jar"))
  .set("spark.cassandra.connection.host", "127.0.0.1")

val ssc = new StreamingContext(sc, Seconds(2))

val stream = TwitterUtils.
  createStream(ssc, None, Nil, storageLevel = StorageLevel.MEMORY_ONLY_SER_2)

val hashTags = stream.flatMap(tweet =>
  tweet.getText.toLowerCase.split(" ").
  filter(tags.contains(Seq("#iphone", "#android"))))

val tagCounts = hashTags.map((_, 1)).reduceByKey(_ + _)

val tagCountsAll = tagCounts.map{
  case (tag, mentions) => (tag, mentions, "ALL")
}

tagCountsAll.saveToCassandra(
```

DATASTAX

```
import com.datastax.spark.connector.streaming._

val sc = new SparkConf()
  .setMaster("spark://127.0.0.1:7077")
  .setAppName("Twitter-Demo")
  .setJars("demo-assembly-1.0.jar"))
  .set("spark.cassandra.connection.host", "127.0.0.1")

val ssc = new StreamingContext(sc, Seconds(2))

val stream = TwitterUtils.
  createStream(ssc, None, Nil, storageLevel = StorageLevel.MEMORY_ONLY_SER_2)

val hashTags = stream.flatMap(tweet =>
  tweet.getText.toLowerCase.split(" ").
  filter(tags.contains(Seq("#iphone", "#android"))))

val tagCounts = hashTags.map((_, 1)).reduceByKey(_ + _)

val tagCountsAll = tagCounts.map{
  case (tag, mentions) => (tag, mentions, "ALL")
}

tagCountsAll.saveToCassandra(
```

DATASTAX

```
val ssc = new StreamingContext(sc, Seconds(2))

val stream = TwitterUtils.
  createStream(ssc, None, Nil, storageLevel = StorageLevel.MEMORY_ONLY_SER_2)

val hashTags = stream.flatMap(tweet =>
  tweet.getText.toLowerCase.split(" ").
  filter(tags.contains(Seq("#iphone", "#android"))))

val tagCounts = hashTags.map((_, 1)).reduceByKey(_ + _)

val tagCountsAll = tagCounts.map{
  case (tag, mentions) => (tag, mentions, "ALL")
}

tagCountsAll.saveToCassandra(
  "demo_ks", "hashtags", Seq("hashtag", "mentions", "interval"))

ssc.start()
ssc.awaitTermination()
```

```
val ssc = new StreamingContext(sc, Seconds(2))

val stream = TwitterUtils.
  createStream(ssc, None, Nil, storageLevel = StorageLevel.MEMORY_ONLY_SER_2)

val hashTags = stream.flatMap(tweet =>
  tweet.getText.toLowerCase.split(" ").
  filter(tags.contains(Seq("#iphone", "#android"))))

val tagCounts = hashTags.map((_, 1)).reduceByKey(_ + _)

val tagCountsByDay = tagCounts.map{
  case (tag, mentions) => (tag, mentions, DateTime.now.toString("yyyyMMdd"))
}

tagCountsByDay.saveToCassandra(
  "demo_ks", "hashtags", Seq("hashtag", "mentions", "interval"))

ssc.start()
ssc.awaitTermination()
```

```
val ssc = new StreamingContext(sc, Seconds(2))

val stream = TwitterUtils.
  createStream(ssc, None, Nil, storageLevel = StorageLevel.MEMORY_ONLY_SER_2)

val hashTags = stream.flatMap(tweet =>
  tweet.getText.toLowerCase.split(" ").
  filter(tags.contains(Seq("#iphone", "#android"))))

val tagCounts = hashTags.map((_, 1)).reduceByKey(_ + _)

val tagCountsAll = tagCounts.map{
  case (tag, mentions) => (tag, mentions, "ALL")
}

tagCountsAll.saveToCassandra(
  "demo_ks", "hashtags", Seq("hashtag", "mentions", "interval"))

ssc.start()
ssc.awaitTermination()
```

Questions ?