Register | Login | Logout | Contact Us

Java-Success.com

Industrial strength Java/JEE Career Companion to open more doors

search here … Go

Home | Java FAQs | 600+ Java Q&As | Career | Tutorials | Member | Why?

Can u Debug? | Java 8 ready? | Top X | Productivity Tools | Judging Experience?

Home › Interview › Hadoop & BigData Interview Q&A › 04: Q27 – Q36 Apache Spark interview questions & answers

04: Q27 – Q36 Apache Spark interview questions & answers

Posted on May 14, 2016 by Arulkumaran Kumaraswamipillai

0 Like Share Tweet 0 G+1 Share

Q27. Where is Apache Spark used in the Hadoop eco system?

A27. Spark is essentially a data processing framework that is faster & more flexible than "Map Reduce". The Spark itself has grown into an eco system with Spark SQL, Spark streaming, Spark UI, GraphX, MLlib, and SparkR. Apache Spark can run on Hadoop clusters, as a standalone system or on the cloud. Spark can be used for fast processing (e.g. transforming sequence files into AVRO or Parquet file formats, reading from HBase, Hive, Cassandra, and any HDFS, etc), for sophisticated analytics (e.g. machine learning & graph algorithm), and for near real time (i.e. NRT)

600+ Full Stack Java/JEE Interview Q&As ♥Free ♦FAQs

open all | close all

⊞ Ice Breaker Interview
⊞ Core Java Interview (
⊞ JEE Interview Q&A (3
⊞ Pressed for time? Jav
⊞ SQL, XML, UML, JSC
⊟ Hadoop & BigData Int
♥ 01: Q1 – Q6 Had
02: Q7 – Q15 Hado
03: Q16 – Q25 Hac
04: Q27 – Q36 Apa
05: Q37 – Q50 Apa
05: Q37-Q41 – Dat
06: Q51 – Q61 HBa
07: Q62 – Q70 HD
⊞ Java Architecture Inte
⊞ Scala Interview Q&As
⊞ Spring, Hibernate, & I
⊞ Testing & Profiling/Sa
⊞ Other Interview Q&A 1
⊞ ▶ Free Java Interviev

streaming of “Discretized Streams or DStreams”. DStreams are defined as sequences of RDD’s.

Q28. Why is Apache Spark favoured over MapReduce as an open source big data processing framework
A28. Spark gives you a comprehensive and unified framework to manage big data processing requirements with near real time (i.e. NRT) latency.

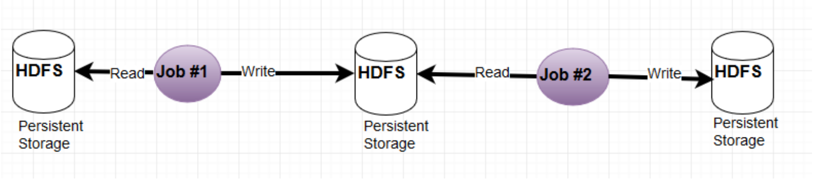
1) In MapReduce, the output data between each step has to be stored in the distributed file system before the next step can begin. This approach can be very slow for iterative tasks due to data replications across nodes & data storage I/O operations. Spark allows you for the steps to run

a) completely in memory for performance,
b) by writing everything to disk to handle large data sets and
c) by writing partially to disk & partially processing from the memory to get the best of both performance & ability to work with large data sets.

You have to look at your data and use cases to assess the memory requirements.

2) In MapReduce, if you want to perform complex processing, you need to string together a series of MapReduce jobs, and execute them in sequence.

1) read data from HDFS -> **2)** apply map and reduce –> **3)** write data back to HDFS –> **1)** read data from HDFS –> **2)** apply map and reduce –> **3)** write data back to HDFS and so on……



MapReduce Jobs

16 Technical Key Areas

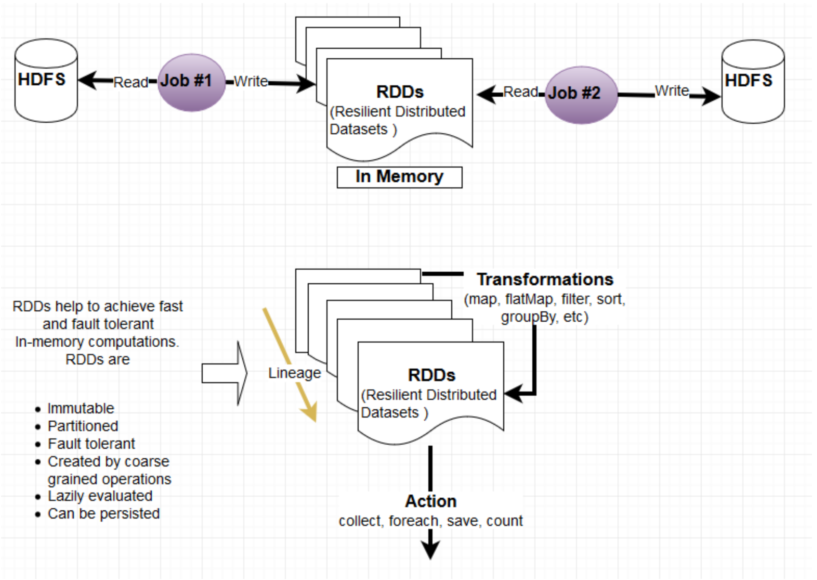
open all | close all
⊞ Best Practice (6)
⊞ Coding (26)
⊞ Concurrency (6)
⊞ Design Concepts (7)
⊞ Design Patterns (11)
⊞ Exception Handling (3
⊞ Java Debugging (21)
⊞ Judging Experience I
⊞ Low Latency (7)
⊞ Memory Managemen
⊞ Performance (13)
⊞ QoS (8)
⊞ Scalability (4)
⊞ SDLC (6)
⊞ Security (13)
⊞ Transaction Managen

80+ step by step Java Tutorials

open all | close all
⊞ Setting up Tutorial (6)
⊞ Tutorial - Diagnosis (2
⊞ Akka Tutorial (9)
⊞ Core Java Tutorials (2
⊞ Hadoop & Spark Tuto
⊞ JEE Tutorials (19)
⊞ Scala Tutorials (1)
⊞ Spring & HIbernate Tu
⊞ Tools Tutorials (19)
⊞ Other Tutorials (45)

Spark allows you to develop complex multi-step pipelines using DAG (i.e. Directed Acyclic Graph) pattern so that different jobs can work with the same data. This makes the development easier, but also makes Spark perform better even if you write everything to disk instead of processing from the memory.

Q29. What is a RDD in Spark?
A29. RDD stands for **R**esilient **D**istributed**D**atasets, which is a collection of fault-tolerant operational elements that run in parallel. The data is immutable & partitioned to run in a distributed manner. RDDs can be cached across computing nodes in a cluster.


HDFS
Read
Job #1
Write
RDDs
(Resilient Distributed Datasets)
In Memory
Read
Job #2
Write
HDFS
RDDs help to achieve fast and fault tolerant In-memory computations. RDDs are
• Immutable
• Partitioned
• Fault tolerant
• Created by coarse grained operations
• Lazily evaluated
• Can be persisted
Lineage
Transformations
(map, flatMap, filter, sort, groupBy, etc)
RDDs
(Resilient Distributed Datasets)
Action
collect, foreach, save, count

RDDs for in-memory computations

JavaSparkContext's parallelize method on a list of integers are copied across the Hadoop cluster (i.e. Data nodes) to form a distributed datasets,

```

SparkConf conf = new SparkConf().setAppName("Sequ
JavaSparkContext sc = new JavaSparkContext(conf);

List<Integer> data = Arrays.asList(1, 2, 3, 4, 5)
JavaRDD<Integer> distData = sc.parallelize(data);

```

and can be operated on in parallel to sum up the elements.

100+ Java pre-interview coding tests

open all | close all
⊞ Can you write code? (
⊞ ♦ Complete the given
⊞ Converting from A to I
⊞ Designing your classe
⊞ Java Data Structures
⊞ Passing the unit tests
⊞ What is wrong with th
⊞ Writing Code Home A
⊞ Written Test Core Jav
⊞ Written Test JEE (1)

How good are your .....?

open all | close all
⊞ Career Making Know-
⊞ Job Hunting & Resum

```

distData.reduce((a, b) -> a + b)

```

You can create a RDD from any storage source like text files, sequence files, avro files, etc.

```

SparkConf conf = new SparkConf().setAppName("Sequ
JavaSparkContext sc = new JavaSparkContext(conf);

JavaRDD<String> distTextFileRDD = sc.textFile("da

```

Q30. What are the different types of RDD operations?
A30. RDD supports two types:

1) Transformations: Create a new dataset from an existing one. For example "**.map**" in the example below is a transformation that extracts values (i.e. _2) from key/value pairs.

2) Actions: Return a value to the driver program after running a computation on the dataset. For example, "**.collect**" in the example below is an action that returns a collection of values.

```

JavaPairRDD<IntWritable, BytesWritable> distSeqFi
         sc.sequenceFile(inputFile.getPath(), IntW
List<String> valuesXml = distSeqFileRDD.map(x ->

```

Q31. What is a "RDD Lineage"?
A31. Spark does not support data replication in memory, hence in an event of any data loss, it is rebuilt using the "RDD Lineage". It is a process of reconstructing lost data partitions.

Q32. How does Spark support development of complex multi-step pipelines?
A32. Spark allows you to develop complex multi-step pipelines using DAG (i.e. Directed Acyclic Graph) pattern so that different jobs can work with the same data. This makes the development easier, but also makes Spark perform better

even if you write everything to disk instead of processing from the memory.

In Spark, a job is associated with a chain of RDD dependencies organized in a direct acyclic graph (DAG) that looks like the following:

Q33. What is a partition in a Spark job?
A33. Partitioning is the process that logically divides units of data to be processed in parallel to speed up data processing. RDDs created in 2 partitions

```

List<Integer> data = Arrays.asList(1, 2, 3, 4, 5,
JavaRDD<Integer> distData = sc.parallelize(data,

```

Q34. How are Spark variables shared across nodes?
A34. When a map or reduce operator is executed on a remote node, it works on separate copies of all the variables used within the operation at a particular node, and any updates to these variables are not propagated back to the driver program. Spark provides 2 approaches to share variables across nodes in a cluster.

1) Accumulators: Variables that can be used to aggregate values from worker nodes back to the driver program.
2) Broadcast variables: Shared variable to efficiently distribute large read-only values to all the worker nodes.

Accumulators

Counting the number of blank lines in a given text input.

```

JavaRDD<String> lines = sc.textFile("data.txt");
final Accumulator<Integer> blankLines = sc.accum
JavaPairRDD<String, Integer> counts = lines.flat
    {
        if ("".equals(line)) {
            blankLines.add(1); // increment the
        }
        return Arrays.asList(line.split(" "));
    }).mapToPair(word -> new Tuple2<String, Integ
```

```
      .reduceByKey((x, y) -> x + y);

System.out.println("Blank lines count: " + blank

```

Broadcast variables

Broadcast the list of words to ignore to all the nodes in a cluster.

```

JavaRDD<String> lines = sc.textFile("data.txt");
final Broadcast<List<String>> wordsToIgnore = sc.

```

Q35. What is a SparkContext?
A35. A "SparkContext" is the main entry point for a Spark job. A "SparkContext" represents the connection to a Spark cluster, and can be used to create RDDs, accumulators and broadcast variables on that cluster.

Create a SparkContext

```

import org.apache.spark.SparkConf;
import org.apache.spark.api.java.JavaPairRDD;
import org.apache.spark.api.java.JavaRDD;
import org.apache.spark.api.java.JavaSparkContex
import org.slf4j.Logger;

//...

  final static JavaSparkContext sc;

  static {
    SparkConf conf =
        new SparkConf().setAppName("Sequence To 
            .set("spark.executor.memory", "1g")
            .set("spark.serializer", "org.apache
    sc = new JavaSparkContext(conf);
  }

//.....

```

Create RDDs

```

JavaRDD<String> lines = sc.textFile("data.txt");

```

Shared variables: Accumulators & Broadcast variables

```

JavaRDD<String> lines = sc.textFile("data.txt");
final Accumulator<Integer> blankLines = sc.accumu
final Broadcast<List<String>> wordsToIgnore = sc.

```

Q36. What is a "Spark streaming"?

A36. Spark is a batch processing platform like Apache Hadoop, and Spark Streaming is a real-time processing tool that runs on top of the Spark engine. Spark streaming is related to Apache Storm, which is the most popular real-time processing platform for Big Data.

The primitive data type for Spark streaming is still RDD's encapsulated by a continuous stream of data known as "Discretized Streams" or DStreams. DStreams are defined as sequences of RDD's. A "DStream" is created from an input source, such as Apache Kafka, or from the transformation of another DStream.

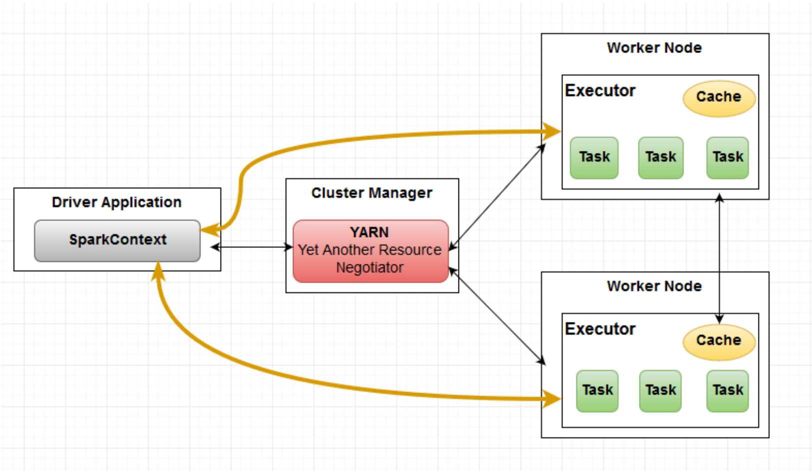
```

SparkConf conf = new SparkConf().setMaster("loca
JavaStreamingContext jsc = new JavaStreamingCont
JavaReceiverInputDStream<String> lines = jsc.soc
JavaDStream<String> words = lines.flatMap(
  new FlatMapFunction<String, String>() {
    @Override public Iterable<String> call(Strin
      return Arrays.asList(x.split(" "));
    }
  });

```

Q37. What is a Spark Executor?

A37. The "Driver Application" creates tasks & schedule them to be run on the "Spark Executors".



SparkContext with Executors executing tasks

Executors are worker nodes' processes in charge of running individual tasks in a given Spark job. Spark Executors are launched at the beginning of a Spark application and typically run for the entire lifetime of an application. Once they have finished running the tasks they send the results to the "Driver Application". "**Spark Executors**" also provide in-memory storage for RDDs that are cached.

```

rdd4.cache()

```

Popular Posts

♦ 11 Spring boot interview questions & answers

828 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

768 views

18 Java scenarios based interview Questions and Answers

400 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

389 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

296 views

♦ 7 Java debugging interview questions & answers

293 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

286 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

280 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

240 views

001B: ♦ Java architecture & design concepts interview questions & answers

202 views

Bio | Latest Posts

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ 03: Q16 – Q25 Hadoop MapReduce interview questions & answers

04: Convert XML file To an Avro File with Apache Spark – writing & reading ›

Posted in Hadoop & BigData Interview Q&A, member-paid

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ Java generics in no time ☀ Top 6 tips to transforming your thinking from OOP to FP ☀ How does a HashMap internally work? What is a hashing function?
☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from “Blah blah” to “Wow”? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category ▼

© Disclaimer

The contents in this Java-Success are copy righted. The author has the right to correct or enhance the current content without any prior notice.

These are general advice only, and one needs to take his/her own circumstances into consideration. The author will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. No guarantees are made regarding the accuracy or usefulness of content, though I do make an effort to be accurate. Links to external sites do not imply endorsement of the linked-to sites.

↑