1. What is RDD Lineage?

Answer: RDD lineage is a graph that Spark uses to keep track of the set of dependencies between different RDDs. It is built as a result of applying transformations to the RDD and creates a [logical execution plan](https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-rdd-lineage.html#logical-execution-plan).

RDD lineage graph can be seen from [RDD.toDebugString](https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-rdd-lineage.html#toDebugString) method

2. What is Spark behavior in following Scenario?

Answer: Consider a cluster of 100 nodes and your application has 10 transformations. One of the node fails or disconnects from the cluster during the execution.

Resubmit failed stages if outputs are lost from the failed/disconnected nodes to other running nodes

3. How do you control parallelism in applications?

Use parallelize, coalesce, repartition

4. What is the difference between map and mapPartitions?

map applies a function to each element in the RDD and return an RDD of the result.

mapPartitions() can be used as an alternative to map() & foreach(). mapPartitions() is called once for each Partition unlike map() & foreach()which is called for each element in the RDD. The main advantage being that, we can do initialization on Per-Partition basis instead of per-element basis(as done by map() & foreach())

5. What are the benefits of Spark Architecture?

Isolation: Applications are completely isolated

Task scheduling per application

Low-overhead:

Task setup cost is that of spawning a thread, not a process, hence it is much faster

Small tasks - mitigate effects of data skew

Sharing data:

Applications cannot share data in memory natively

Use an external storage service like Tachyon

Resource allocation: Static process provisioning for executors, even without active tasks