Spark Interview – Simple & Clear Answers (Manager■Friendly)

This document contains simplified, clear, and interview■ready answers to Spark questions. You can explain confidently to senior managers and demonstrate strong conceptual knowledge.

1. How do you run a Spark application?

- We run Spark applications using the spark-submit command.
- spark-submit sends the application code, configurations, and dependencies to the Spark cluster.
- Spark can run in different modes: Local (for testing), Standalone, YARN, Mesos, and Kubernetes.
- Example: spark-submit --master yarn --deploy-mode cluster app.jar

2. What is spark-submit and its important options?

- spark-submit is the command-line tool used to launch Spark applications.
- Key options: --master (cluster type), --deploy-mode (client/cluster), --executor-memory, --executor-cores, --num-executors.
- These options control performance, resource usage, and execution behavior.

3. How do you know how many stages are in a Spark job?

- Use the Spark UI (Stages tab).
- Each stage represents a group of tasks executed together.
- Stages are mainly created because of shuffle (wide transformations).

4. How do you increase parallelism in Spark?

- Increase the number of partitions using repartition().
- Increase executors and cores using spark-submit options.
- Tune configurations like spark.sql.shuffle.partitions.
- Handle data skew to avoid slow tasks.

5. How does Spark decide the number of stages?

- Spark builds a DAG (Directed Acyclic Graph) of transformations.
- Each wide transformation (shuffle) creates a new stage.
- Narrow transformations run in the same stage.

6. How do you make a Spark job run faster?

- Optimize partitions and resource allocation.
- Use cache() for reused DataFrames.
- Avoid unnecessary shuffles.

- Use broadcast joins for small lookup tables.
- Check execution plans using explain().

7. What file formats have you used?

- CSV – for small or external data exchange.
- JSON – for semi-structured API data.
- Parquet – for large analytical datasets (best performance).
- ORC and Avro – for schema-based storage and evolution.

8. Difference between Parquet and CSV?

- Parquet is columnar, CSV is row-based.
- Parquet supports compression and predicate pushdown.
- Parquet is faster for analytics; CSV is simple but slow for big data.

9. How does partitioning improve performance?

- Partitioning reduces data scanning using partition pruning.
- Enables parallel processing.
- Reduces shuffle and improves query speed.

10. What are narrow and wide transformations?

- Narrow: map, filter – no shuffle, faster.
- Wide: groupBy, join – shuffle required, slower.
- Wide transformations create new stages.

11. Explain Spark architecture.

- Driver: Controls job execution and scheduling.
- Cluster Manager: Allocates resources.
- Executors: Run tasks and store data.
- SparkSession is the main entry point.

12. Difference between RDD and DataFrame?

- RDD is low-level and unstructured.
- DataFrame is high-level and optimized.
- DataFrames use Catalyst optimizer and are faster.
- RDD is used only for special low-level needs.

13. What is a partition in Spark?

- A partition is a chunk of data processed by one task.
- More partitions = more parallelism (up to available cores).
- Partitions are the base unit of Spark execution.

14. How do you get schema-wise table count using Spark?

- Use spark.catalog.listDatabases() to get schemas.
- Use spark.catalog.listTables(schema) to get tables.
- Count tables per schema and display results.