

# 1. Spark Basics

- **What is Apache Spark?** – A distributed computing engine for big data, faster than Hadoop MapReduce.
- **Core Components:**
  - Spark Core (RDDs, DAG execution)
  - Spark SQL (structured data)
  - Spark Streaming (real-time)
  - MLlib (machine learning)
  - GraphX (graph processing)

**Code Example (Scala – Word Count using RDDs):**

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

object WordCount {
  def main(args: Array[String]): Unit = {
    val spark = SparkSession.builder
      .appName("Word Count")
      .master("local[*]")
      .getOrCreate()

    val sc = spark.sparkContext
    val textFile = sc.textFile("sample.txt")

    val counts = textFile.flatMap(line => line.split(" "))
      .map(word => (word, 1))
      .reduceByKey(_ + _)

    counts.collect().foreach(println)
    spark.stop()
  }
}
```

---

## 2. Spark RDD vs DataFrame vs Dataset

- **RDD:** Low-level, untyped distributed collection.

- **DataFrame:** High-level, schema-based, optimized by Catalyst.
- **Dataset:** Type-safe, combines RDD benefits with DataFrame optimizations.

#### Example: DataFrame Operations

```
val df = spark.read.option("header", "true").csv("employees.csv")
df.select("name", "salary").where("salary > 50000").show()
```

---

### 3. Spark SQL

- Query structured data with SQL.
- Integrated with Hive metastore.
- Catalyst optimizer for query planning.

#### Example:

```
df.createOrReplaceTempView("employees")
val highSalary = spark.sql("SELECT name, salary FROM employees WHERE
salary > 50000")
highSalary.show()
```

---

### 4. Spark Transformations & Actions

- **Transformations:** lazy operations (`map`, `filter`, `flatMap`, `join`).
- **Actions:** trigger computation (`collect`, `count`, `saveAsTextFile`).

#### Example:

```
val numbers = sc.parallelize(1 to 10)
val even = numbers.filter(_ % 2 == 0)
println("Count of evens: " + even.count())
```

---

### 5. Spark Streaming (Structured Streaming)

- Real-time data from Kafka, sockets, files.
- Micro-batching model.

### Example: Streaming from Socket

```
val lines = spark.readStream
    .format("socket")
    .option("host", "localhost")
    .option("port", 9999)
    .load()

val words = lines.as[String].flatMap(_.split(" "))
val counts = words.groupBy("value").count()

counts.writeStream
    .outputMode("complete")
    .format("console")
    .start()
    .awaitTermination()
```

---

## 6. Spark in ETL

- **Ingestion:** Kafka, HDFS, S3, JDBC
- **Transformation:** Cleaning, aggregation, joins, windowing
- **Loading:** Write back to DB, DW, S3

### Pipeline Example:

```
val input = spark.read.json("s3://bucket/input/")
val transformed = input.filter("age > 25").groupBy("city").count()
transformed.write.mode("overwrite").parquet("s3://bucket/output/")
```

---

## 7. Spark Performance Tuning

- Use **partitioning & caching** effectively.
  - Avoid `collect()` on large datasets.
  - Optimize joins with `broadcast`.
  - Tune Spark configs (`spark.sql.shuffle.partitions`, `executor memory`).
-

## 8. Spark Cluster & Deployment

- Modes: Local, Standalone, YARN, Kubernetes.
  - Submitting apps: `spark-submit`
  - Monitoring: Spark UI (<http://localhost:4040>)
- 

⚡ If you'd like, I can:

1. Build you a **step-by-step Spark ETL project plan** (with GitHub repos).
2. Create **cheat sheets** for Spark (RDD, SQL, Streaming, Tuning).
3. Provide **interview Q&A + coding challenges**.

👉 Do you want me to expand into **practical ETL project (Kafka → Spark → Data Lake)** first, or give you a **Spark cheat sheet with quick commands**?

**You said:**

quiz on spark 80 questions , beginners, intermediate, advanced and expert

**ChatGPT said:**

Perfect 👍 You want a **comprehensive Spark quiz** – 80 questions across **beginner, intermediate, advanced, and expert levels**. I'll break them into **20 questions per level** so you can test your progression.

Here's the full quiz:

---



## Apache Spark Quiz – 80 Questions

---

### Beginner Level (20 Questions)

1. What does Apache Spark primarily provide?
  - a) Real-time OS
  - b) Distributed data processing framework
  - c) Database management system
  - d) Web framework
2. Which language is NOT supported by Spark out-of-the-box?
  - a) Python
  - b) Scala

- c) Ruby
  - d) Java
3. What does RDD stand for?
- a) Resilient Distributed Dataset
  - b) Real-time Distributed Data
  - c) Reliable Data Distribution
  - d) Remote Data Definition
4. Which of these is an **action** in Spark?
- a) `map`
  - b) `filter`
  - c) `collect`
  - d) `flatMap`
5. Which Spark component allows SQL-like queries?
- a) Spark Core
  - b) Spark SQL
  - c) Spark MLlib
  - d) Spark Streaming
6. In Spark, transformations are:
- a) Eager
  - b) Lazy
  - c) Parallel only
  - d) Blocking
7. Spark stores data in-memory by default. True or False?
8. What is the default cluster manager in Spark?
- a) Mesos
  - b) YARN
  - c) Standalone
  - d) Kubernetes
9. Which API is recommended for structured data?
- a) RDD
  - b) DataFrame
  - c) GraphX
  - d) MLlib
10. Which file format does Spark NOT natively support?
- a) Parquet
  - b) ORC
  - c) Avro
  - d) MDB
11. What is Spark Shell written in?
- a) Python
  - b) Scala
  - c) Java
  - d) R

12. Which Spark component handles graphs?
- a) Spark GraphX
  - b) Spark SQL
  - c) Spark MLlib
  - d) Spark Streaming
13. Which of these is a **lazy transformation**?
- a) `count()`
  - b) `map()`
  - c) `reduce()`
  - d) `collect()`
14. What does Spark use for fault tolerance?
- a) Replication
  - b) Lineage (DAG)
  - c) Checkpoints only
  - d) RAID
15. Which of these creates a DataFrame in Spark?
- a) `spark.read.csv()`
  - b) `spark.textFile()`
  - c) `sc.parallelize()`
  - d) `df.createOrReplaceTempView()`
16. Default shuffle partition value in Spark SQL?
- a) 100
  - b) 200
  - c) 256
  - d) 512
17. Spark runs on Hadoop using:
- a) HDFS only
  - b) YARN
  - c) MapReduce engine
  - d) Pig
18. Which ML library is integrated with Spark?
- a) TensorFlow
  - b) MLlib
  - c) Scikit-learn
  - d) Keras
19. Which method writes DataFrame in Parquet format?
- a) `df.save("path")`
  - b) `df.write.parquet("path")`
  - c) `df.write.csv("path")`
  - d) `df.parquet("path")`
20. Which Spark UI port shows application info locally?
- a) 7077

- b) 8080
  - c) 4040
  - d) 50070
- 

## Intermediate Level (20 Questions)

- 21. Catalyst optimizer is used in:
  - a) Spark Streaming
  - b) Spark SQL
  - c) Spark Core
  - d) GraphX
- 22. Tungsten project in Spark is related to:
  - a) Memory and CPU optimization
  - b) Streaming ingestion
  - c) Data replication
  - d) Fault tolerance
- 23. In Spark, Broadcast variables are used to:
  - a) Cache intermediate RDDs
  - b) Share read-only data across nodes
  - c) Replicate data in executors
  - d) Store shuffle files
- 24. Accumulators are:
  - a) Immutable RDDs
  - b) Read-only shared variables
  - c) Write-only variables for aggregation
  - d) Optimizers
- 25. Which join type is NOT directly supported in Spark SQL?
  - a) Inner
  - b) Outer
  - c) Semi
  - d) Cross
- 26. Spark MLlib's DataFrame-based API replaced:
  - a) Old RDD-based MLlib
  - b) GraphX
  - c) TensorFlow integration
  - d) Catalyst
- 27. What is the default serialization mechanism in Spark?
  - a) Avro
  - b) Java serialization
  - c) Kryo
  - d) JSON

28. Structured Streaming uses:
- a) Continuous processing model
  - b) Micro-batch model
  - c) MapReduce jobs
  - d) Graph processing
29. Checkpointing in Spark is used for:
- a) DAG optimization
  - b) Fault tolerance and state recovery
  - c) Shuffle partitioning
  - d) In-memory caching
30. Which storage level consumes least memory?
- a) MEMORY\_ONLY
  - b) MEMORY\_AND\_DISK
  - c) DISK\_ONLY
  - d) MEMORY\_ONLY\_SER
31. What does `repartition()` do in Spark?
- a) Reduces partitions only
  - b) Increases partitions only
  - c) Increases/decreases partitions with shuffle
  - d) Sorts partitions
32. Coalesce vs Repartition: Which causes shuffle?
- a) Coalesce
  - b) Repartition
  - c) Both
  - d) Neither
33. Which API provides type safety in Spark?
- a) RDD
  - b) DataFrame
  - c) Dataset
  - d) GraphX
34. Predicate pushdown in Spark improves:
- a) Caching
  - b) Filtering at data source
  - c) DAG visualization
  - d) Parallelism
35. Spark Streaming can consume data from:
- a) Kafka
  - b) Flume
  - c) HDFS
  - d) All of the above
36. Spark SQL integrates with Hive using:
- a) HiveContext
  - b) HiveMetastoreClient



- c) HiveJDBC
  - d) Beeline
37. Which is better for small reference data joins?
- a) Sort-Merge Join
  - b) Broadcast Join
  - c) Shuffle Hash Join
  - d) Nested Loop Join
38. Which mode executes Spark locally?
- a) `master=yarn`
  - b) `master=local[*]`
  - c) `master=standalone`
  - d) `master=cluster`
39. Spark's lazy evaluation defers execution until:
- a) Transformation
  - b) Action
  - c) Serialization
  - d) DAG build
40. Which format is columnar and compressed?
- a) JSON
  - b) CSV
  - c) Parquet
  - d) Text
- 

## Advanced Level (20 Questions)

41. In Spark, lineage graphs track:
- a) SQL execution plans
  - b) RDD transformations for fault recovery
  - c) Streaming windows
  - d) Shuffle partitions
42. Tungsten execution engine includes:
- a) Code generation
  - b) Vectorized processing
  - c) Cache-aware computation
  - d) All of the above
43. Which shuffle manager was introduced in Spark 3.x?
- a) Sort-based shuffle
  - b) Tungsten shuffle
  - c) Push-based shuffle
  - d) Hash shuffle

44. Adaptive Query Execution (AQE) adjusts:
- a) Partition sizes and join strategies at runtime
  - b) RDD persistence
  - c) SQL syntax
  - d) Kafka offsets
45. Which join is most efficient for skewed data?
- a) Sort-Merge
  - b) Shuffle Hash
  - c) Skew Join handling with Broadcast
  - d) Cartesian
46. Window functions in Spark SQL are evaluated:
- a) Before filtering
  - b) After groupBy
  - c) After projection
  - d) On partitions with ordering
47. Dynamic allocation in Spark adjusts:
- a) Executor count based on workload
  - b) Shuffle memory only
  - c) RDD persistence
  - d) Spark UI logs
48. The Spark driver program runs on:
- a) Worker node
  - b) Master node
  - c) Client or cluster depending on deploy mode
  - d) HDFS NameNode
49. Delta Lake provides:
- a) ACID transactions on data lakes
  - b) Graph processing
  - c) ML training pipelines
  - d) Real-time dashboards
50. Spark on Kubernetes requires:
- a) YARN
  - b) Pod scheduling for executors
  - c) Mesos
  - d) Flink runtime
51. Which function enables schema evolution?
- a) mergeSchema
  - b) schemaUpdate
  - c) evolveSchema
  - d) withSchema
52. Spark Structured Streaming supports watermarking for:
- a) Event-time handling & late data
  - b) Cluster monitoring

- c) Schema inference
  - d) Catalyst optimization
53. Which command persists DataFrame in memory & disk?
- a) `df.cache()`
  - b) `df.persist(StorageLevel.MEMORY_AND_DISK)`
  - c) Both
  - d) None
54. Whole-stage code generation avoids:
- a) Java Virtual Calls overhead
  - b) Shuffle
  - c) Serialization
  - d) Network transfer
55. Spark's execution DAG consists of:
- a) Stages & tasks
  - b) Jobs & queries only
  - c) Workers & executors only
  - d) Partitions & HDFS blocks
56. Z-order clustering in Delta Lake helps with:
- a) Data skipping in queries
  - b) Replication
  - c) Backup
  - d) DAG optimization
57. In Spark ML pipelines, transformers:
- a) Convert one DataFrame into another
  - b) Train ML models
  - c) Store features
  - d) Tune hyperparameters
58. Spark GraphX uses:
- a) Property Graph model
  - b) Triple store
  - c) RDF model
  - d) Key-Value pairs
59. Catalyst optimizer uses:
- a) Rule-based & cost-based optimization
  - b) Only cost-based
  - c) Only rule-based
  - d) No optimization
60. In Spark Streaming, stateful operations require:
- a) Checkpointing
  - b) Watermarking
  - c) Tungsten execution
  - d) Catalyst
-

## Expert Level (20 Questions)

61. AQE optimizes skew joins by:
  - a) Splitting skewed partitions
  - b) Replicating data across executors
  - c) Disabling shuffle
  - d) DAG pruning
62. In Spark, barrier execution mode is used for:
  - a) Deep learning distributed training
  - b) SQL queries
  - c) Graph processing
  - d) Small file optimization
63. Photon engine in Databricks replaces:
  - a) JVM execution with C++ vectorized engine
  - b) Catalyst optimizer
  - c) HDFS replication
  - d) Spark Streaming
64. Hyperspace indexing improves:
  - a) Data skipping & query performance
  - b) Executor allocation
  - c) Catalyst rules
  - d) MLlib
65. Kudu integration with Spark provides:
  - a) Real-time updates with columnar storage
  - b) MLlib training pipelines
  - c) SQL parsing
  - d) GraphX models
66. Which Spark config controls shuffle partitions in AQE?
  - a) `spark.sql.shuffle.partitions`
  - b) `spark.shuffle.compress`
  - c) `spark.executor.instances`
  - d) `spark.dynamicAllocation.enabled`
67. Iceberg tables in Spark allow:
  - a) Hidden partitioning & time travel
  - b) Broadcast joins
  - c) AQE optimization
  - d) Whole-stage codegen
68. Project Tungsten's unsafe row optimizes:
  - a) Off-heap memory storage
  - b) Shuffle data replication
  - c) Streaming state recovery
  - d) Serialization format

69. Continuous Processing Mode in Spark achieves latency:
- a) 100 ms
  - b) 1 second
  - c) 10 ms
  - d) 10 seconds
70. Which Spark storage level is best for iterative ML?
- a) MEMORY\_ONLY\_SER
  - b) MEMORY\_AND\_DISK
  - c) MEMORY\_ONLY
  - d) DISK\_ONLY
71. Which optimizer feature was added in Spark 3?
- a) Cost-based optimizer
  - b) Adaptive Query Execution
  - c) Rule-based optimizer
  - d) Whole-stage codegen
72. Photon (Databricks) supports:
- a) ANSI SQL compliance
  - b) Columnar execution
  - c) Vectorized I/O
  - d) All of the above
73. In Spark, speculative execution helps with:
- a) Straggler task mitigation
  - b) Catalyst optimization
  - c) Serialization speed
  - d) Shuffle compression
74. Which Spark library supports AutoML?
- a) MLlib
  - b) MLflow
  - c) Delta
  - d) GraphX
75. The metadata layer in Delta Lake is stored in:
- a) JSON transaction log (`_delta_log`)
  - b) Hive metastore only
  - c) Avro log
  - d) Kafka
76. Spark RAPIDS accelerates:
- a) SQL queries on GPUs
  - b) Shuffle on CPUs
  - c) Kafka ingestion
  - d) Streaming windows
77. In Spark, Kubernetes dynamic allocation requires:
- a) Shuffle tracking service
  - b) HDFS checkpointing

- c) Spark SQL Catalyst
- d) GraphX

78. Which Spark config enables AQE?

- a) `spark.sql.adaptive.enabled=true`
- b) `spark.shuffle.spill=true`
- c) `spark.sql.execution.arrow.enabled=true`
- d) `spark.dynamicAllocation.enabled=true`

79. Hudi integrates with Spark to provide:

- a) Incremental data pipelines with ACID
- b) Graph analytics
- c) ML pipelines
- d) GPU acceleration

80. In Spark 3.x, ANSI SQL compliance ensures:

- a) Standard error handling and SQL semantics
- b) Catalyst replacement
- c) Faster shuffle
- d) TensorFlow integration