**1. Intro to HDFS and Common Terminology**

* **HDFS**: Hadoop Distributed File System – designed for reliable, scalable storage of large datasets.
* **NameNode**: Master node that manages file system metadata (file names, blocks, locations).
* **DataNode**: Worker node that stores actual data blocks.
* **Block**: A fixed-size chunk of a file (default: 128 MB or 256 MB).
* **Replication**: Copies of blocks for fault tolerance.
* **Cluster**: Group of connected machines (nodes) running HDFS.

**2. Why HDFS**

* Handles **massive data volumes** across distributed machines.
* **Fault-tolerant** via replication.
* **Highly scalable** and cost-effective.
* Supports **parallel processing** (MapReduce, Spark).
* Optimized for **high throughput, not low-latency access**.

**3. HDFS Architecture**

* **Master-slave architecture**.
* **NameNode** stores metadata (in RAM), no actual data.
* **DataNodes** store the actual file blocks.
* **Secondary NameNode** periodically merges edit logs with the file system image (for backup and recovery).
* Client communicates with NameNode for metadata and directly with DataNodes for data transfer.

**4. Blocks in HDFS**

* Files split into large blocks (default: 128 MB/256 MB).
* Stored across multiple DataNodes.
* Large size reduces metadata load and improves throughput.
* Each block is identified by a unique ID.

**5. Replication Factor in HDFS**

* Default is **3**: one local, one on a different rack, and one on the same rack.
* Ensures **fault tolerance and data availability**.
* If a DataNode fails, missing replicas are automatically recreated.

**6. Rack Awareness in HDFS**

* HDFS is rack-aware to improve **fault tolerance and network efficiency**.
* Data is placed across different **racks** to protect against rack-level failures.
* At least one replica is stored in a different rack to ensure data availability if an entire rack fails.

Top of Form

Bottom of Form