**How data is stored in HDFS**

**Blocks in HDFS**

* **HDFS splits large files into fixed-size blocks** (default: **128MB or 256MB**).
* Each block is assigned a **unique ID** and stored across multiple **DataNodes**.
* Blocks are the **smallest unit of data storage** in HDFS.

**Example:**

If you upload a **600MB** file and the block size is **128MB**, it will be split into:  
📦 **Block 1** → 128MB  
📦 **Block 2** → 128MB  
📦 **Block 3** → 128MB  
📦 **Block 4** → 128MB  
📦 **Block 5** → 88MB (remaining size)

**DataNodes (Storage Nodes)**

* DataNodes **store the actual blocks**.
* They **send heartbeats** to the NameNode to indicate they are active.
* If a DataNode fails, HDFS automatically **re-replicates missing blocks** from other nodes.

**How Blocks are Stored in DataNodes:**

A file’s blocks are **distributed randomly across different DataNodes** for load balancing.

For example:  
📦 **Block 1 → DataNode 1**  
📦 **Block 2 → DataNode 3**  
📦 **Block 3 → DataNode 2**  
📦 **Block 4 → DataNode 4**

**Racks in HDFS (Rack Awareness)**

* A **Rack** is a group of DataNodes **physically located together** in a data center.
* HDFS follows **Rack Awareness** to place blocks intelligently:
  + Ensures **fault tolerance** by **storing copies in different racks**.
  + **Minimizes data loss** if an entire rack fails.

**Example of Block Replication in Different Racks:**

(Default Replication Factor = 3)

| **Block** | **Replica 1 (Rack 1)** | **Replica 2 (Rack 2)** | **Replica 3 (Rack 1)** |
| --- | --- | --- | --- |
| Block 1 | DataNode 1 | DataNode 5 | DataNode 2 |
| Block 2 | DataNode 3 | DataNode 6 | DataNode 4 |
| Block 3 | DataNode 2 | DataNode 7 | DataNode 1 |

HDFS **stores at least one copy in a different rack** to reduce data loss in case of a rack failure.

**NameNode (Master Node)**

* Stores **metadata** (block locations, permissions, and file system hierarchy).
* **Does NOT store actual data**, only a **mapping of which block is stored in which DataNode**.

**How the NameNode Helps in Data Storage:**

1. A client uploads a file.
2. **NameNode splits the file into blocks** and assigns DataNodes for storage.
3. DataNodes **store blocks and send heartbeats** to the NameNode.
4. **Replication ensures fault tolerance** across racks.

**Replication for Fault Tolerance**

* The default **replication factor = 3** (can be configured).
* Blocks are stored on **different DataNodes across racks**.
* If a DataNode fails, **HDFS automatically creates new replicas** from other copies.

**Example of Replication Across Racks:**

📦 **Block 1** → Stored in: **Rack 1 (DataNode 1), Rack 2 (DataNode 5), Rack 1 (DataNode 2)**  
📦 **Block 2** → Stored in: **Rack 2 (DataNode 3), Rack 1 (DataNode 4), Rack 2 (DataNode 6)**