HADOOP: Part I

Introduction to Big Data and Data Analytics
(20B12CS333)

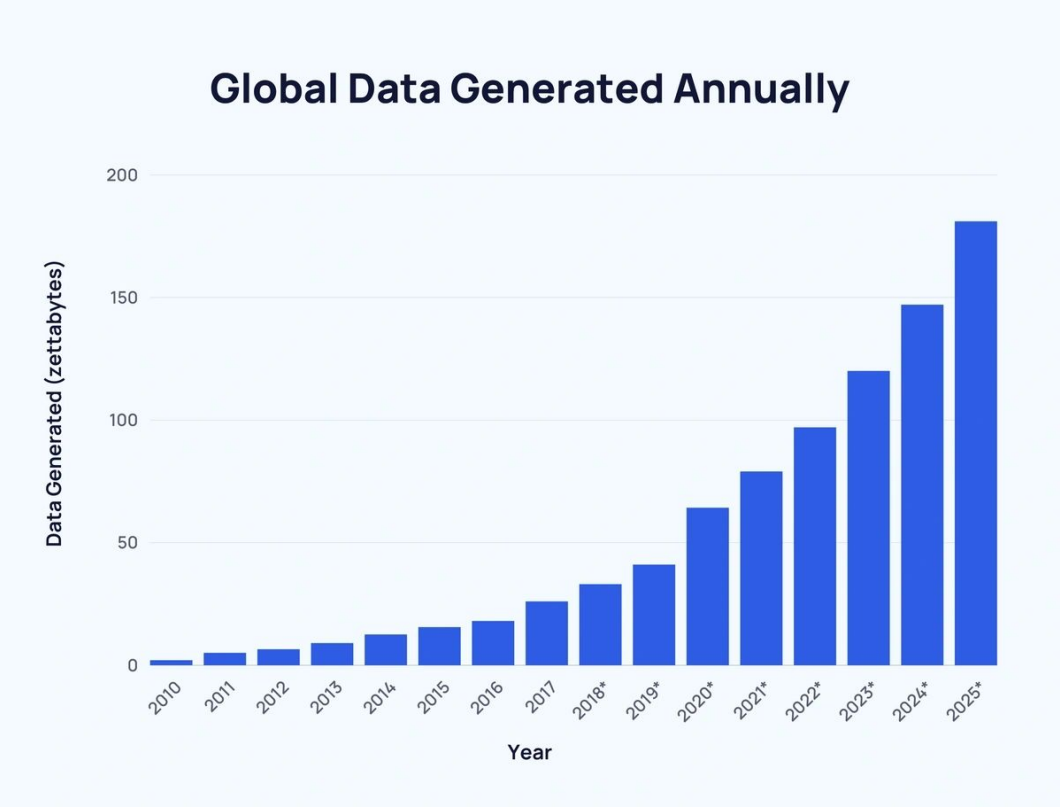
Noor Mohammad
Assistant Professor, C.S.E. & I.T.
noor.mohammad@mail.jiit.ac.in
Cabin no: 3016

Department of Computer Science & Engineering and Information Technology
Jaypee Institute of Information Technology (JIIT), Sector 128, Noida

Problems with the Big Data

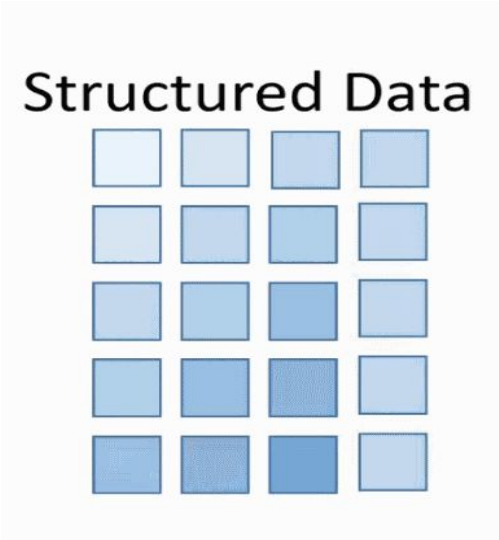
Problem 1: Big Data Storage Problem

- Approximately 402.74 million terabytes (or about 0.4 zettabytes) of data are generated each day.
- The amount of data generated annually has grown year-over-year since 2010.
- In fact, it is estimated that 90% of the world's data was generated in the last two years alone.

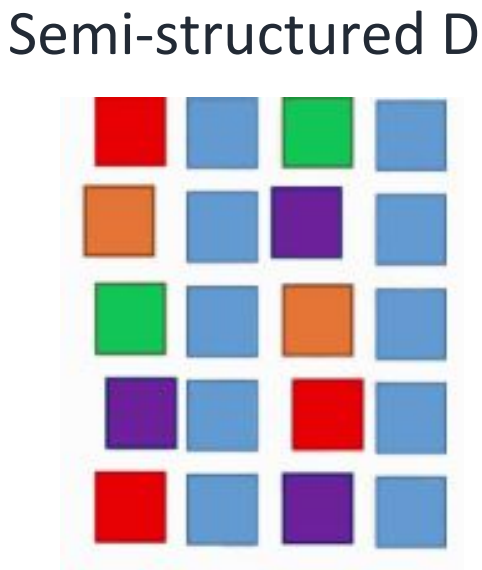


Problems with the Big Data continued…..

Problem 2: Big Data has complex structure

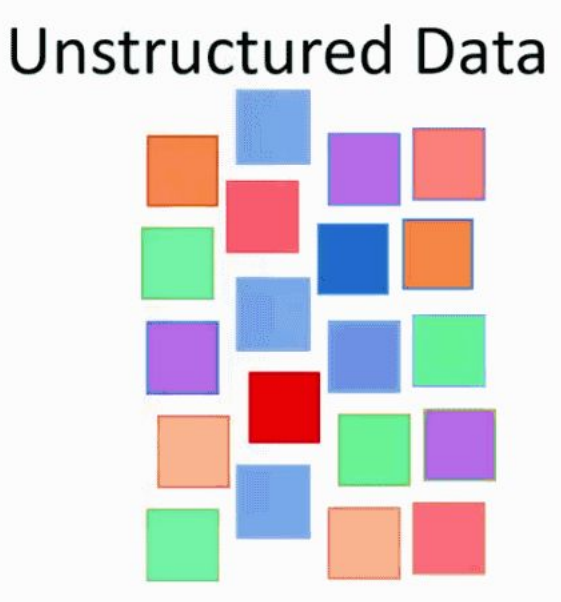


- Data is in organised way.
- Schema for data is fixed.
- For example: RDBMS data.



- Data is partially organized.
- Lacks of formal structure of data model.
- For example: XML, JSON, etc.

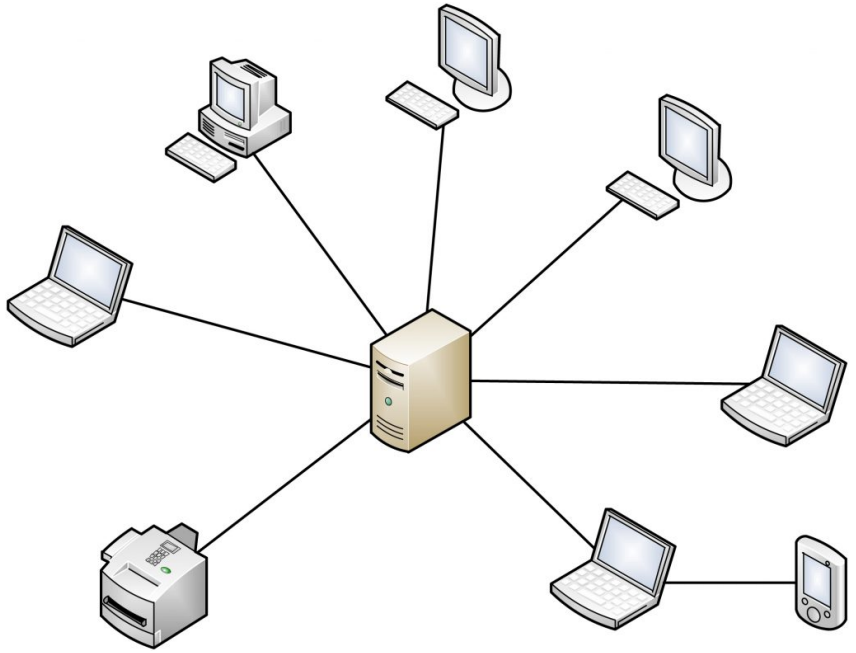
- Unorganized data.
- Unknown schema.
- For example: Multimedia files, audio, video, images, etc.



Problems with the Big Data continued…..

Problem 3: Big Data Processing

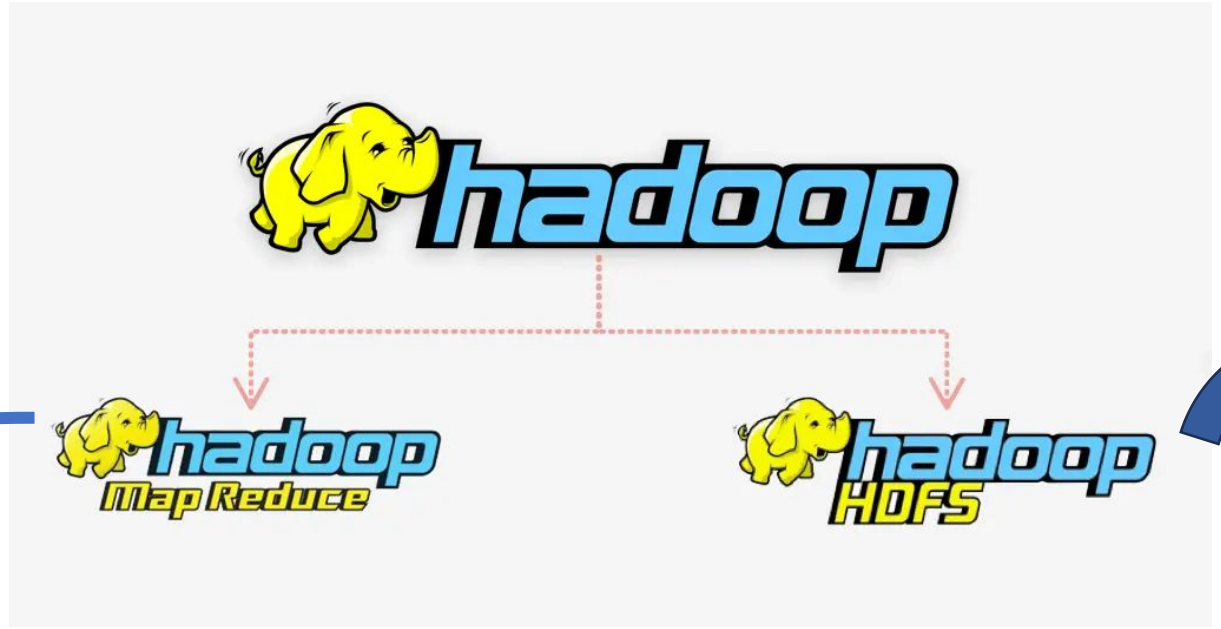
- Data is growing at much faster rate than disk read/write speed.
- Bringing huge amount of data to computation unit becomes a *bottleneck.*



What is the solution?

Hadoop – as a Solution

Hadoop is a framework that allows us to store and process large datasets in parallel and distributed fashion.



Allows parallel processing of the data stored in HDFS.

Allows to dump any kind of data across the cluster.

Hadoop

- Hadoop is a framework for running jobs on clusters of computers that provides a good abstraction of the underlying hardware and software.

"Stripped to its core, the tools that Hadoop provides for building distributed systems—for data storage, data analysis, and coordination—are simple. If there's a common theme, it is about raising the level of abstraction—to create building blocks for programmers who just happen to have lots of data to store, or lots of data to analyze, or lots of machines to coordinate, and who don't have the time, the skill, or the inclination to become distributed systems experts to build the infrastructure to handle it."

White T. Hadoop: The Definitive Guide. O'Reilly, 1988

Hadoop continued….

- Hadoop is an open-source project of the Apache Software Foundation.
- The project was created to facilitate computations involving massive amounts of data.
- Its core components are implemented in Java. initially released in 2006.

Hadoop's features addressing the challenges of Big Data:

- Scalability
- Fault tolerance
- High availability
- Distributed cache/data locality
- Cost-effectiveness as it does not need high-end hardware
- Provides a good abstraction of the underlying hardware
- Easy to learn
- Data can be queried trough SQL-like endpoints (Hive, Cassandra)

Hadoop's distinguishing features

- **Fault tolerance:** The ability to withstand hardware or network failures
- **High availability:** This refers to the system minimizing downtimes by eliminating single points of failure.
- **Data locality**: Task are run on the node where data are located, in order to reduce the cost of moving data around.

HDFS

- HDFS has a master/slave architecture.
- Architecture consists of three major components - the Client, the Master node and the slave nodes.

Master node

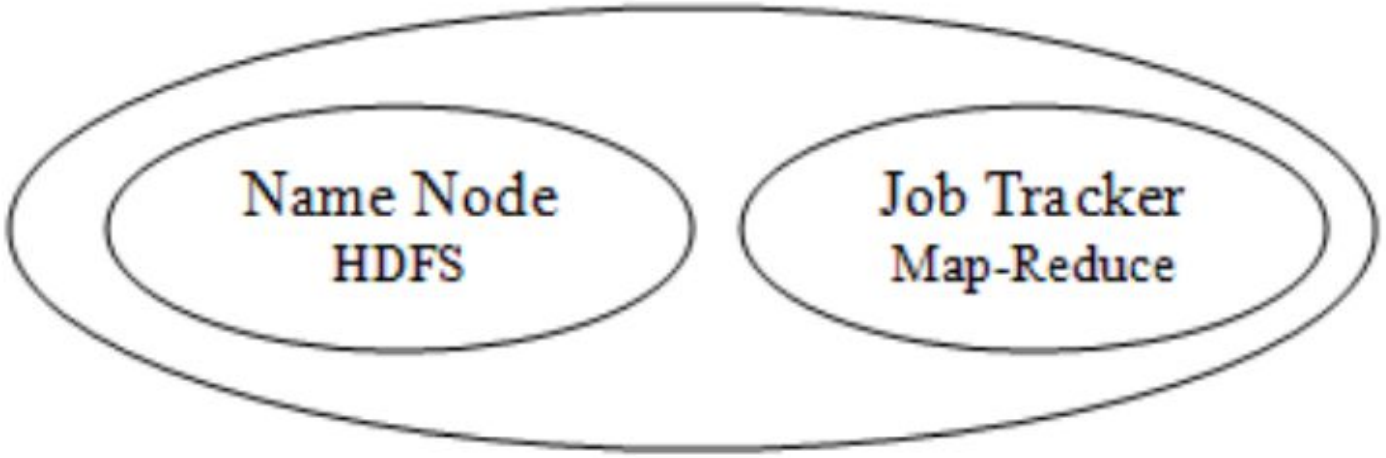
Responsible for two functionalities:

1. Distributing data obtained form the client to the HDFS
2. Assigning tasks to Map-Reduce layer.

Master node

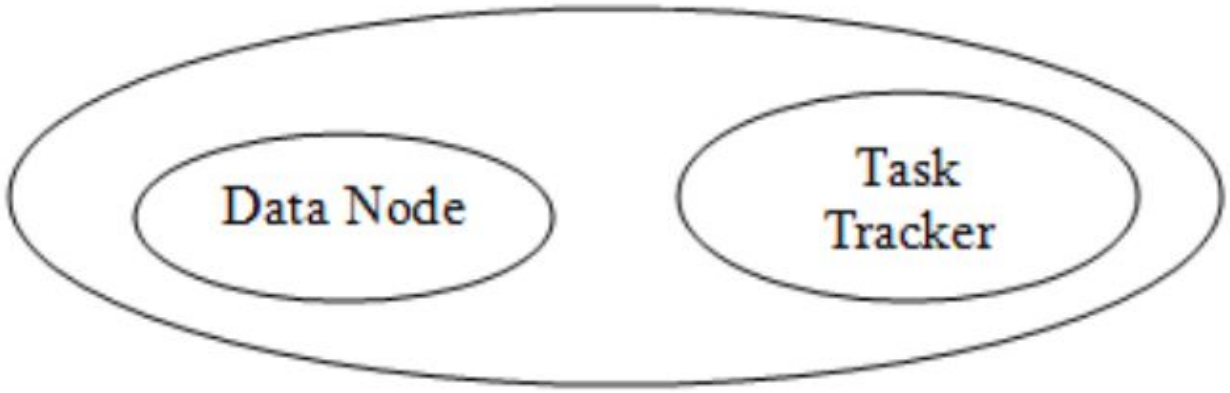
These functions are handled by two different nodes:

- **Name Node:** helps in coordinating HDFS
- **Job Tracker:** helps in coordinating parallel execution of Map-Reduce.



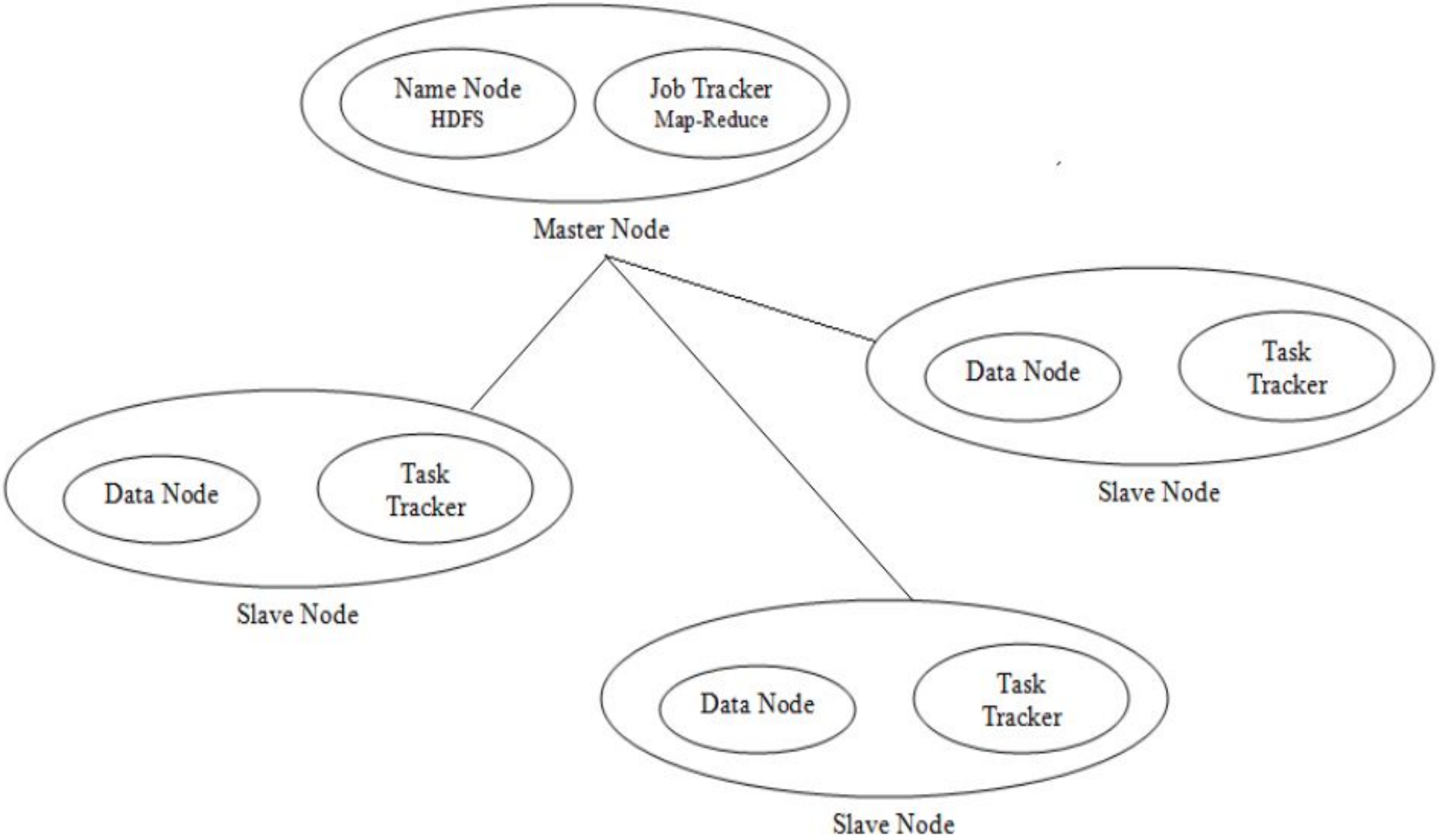
Slave node

- Slave node consists of Data Node and Task Tracker.
- There are number of slave nodes in a Hadoop cluster.
- These nodes communicate and receive instructions for the Master nodes and perform the assigned tasks.



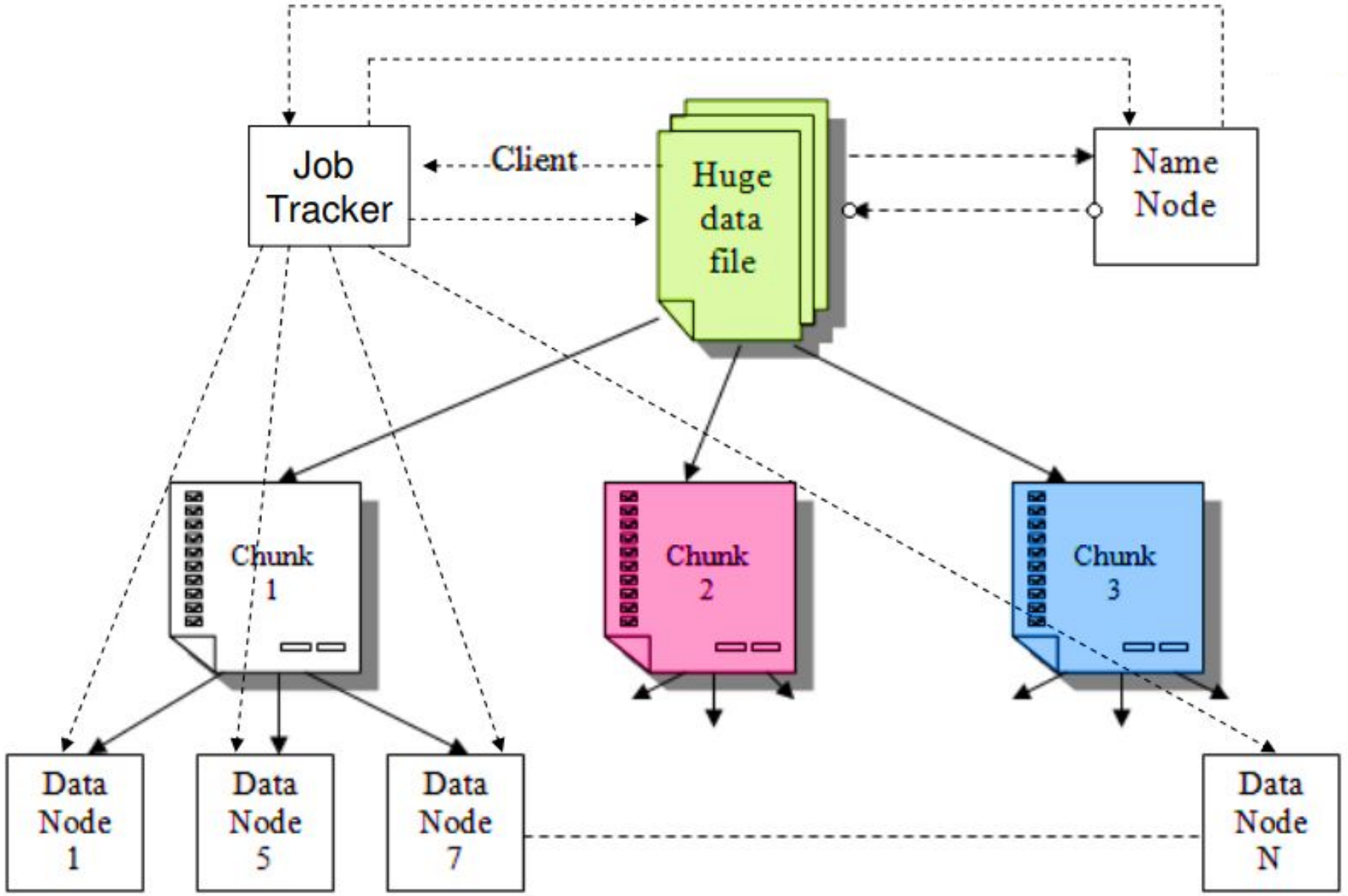
HDFS continued…

Master-Slave Architecture



HDFS continued…

Working: HDFS



HDFS: Name Node

- An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients.
- HDFS exposes a file system namespace and allows user data to be stored in files.
- Executes file system namespace operations like opening, closing, and renaming files and directories.
- It is the arbitrator and repository for all HDFS metadata.

HDFS: Name Node's Role

- Keeps track of What chucks belong to a file and which Data Node holds its copy.
- Cluster's storage capacity.
- Making sure each chunk of file has the minimum number of copies in the cluster as required.
- Directs clients for write or read operation.
- Schedule and execute Map Reduce jobs.

Name Node : Job Tracker

- Client submitted Map-Reduce logic.
- Responsible for scheduling the task to the slave nodes.
- Job Tracker consults the Name Node and assigns the task to the nodes which has the data on which this task would be performed.

HDFS: Data Node

- Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes.
- It also determines the mapping of blocks to DataNodes.
- Responsible for serving read and write requests from the file system's clients.
- Also, perform block creation, deletion, and replication upon instruction from the NameNode.

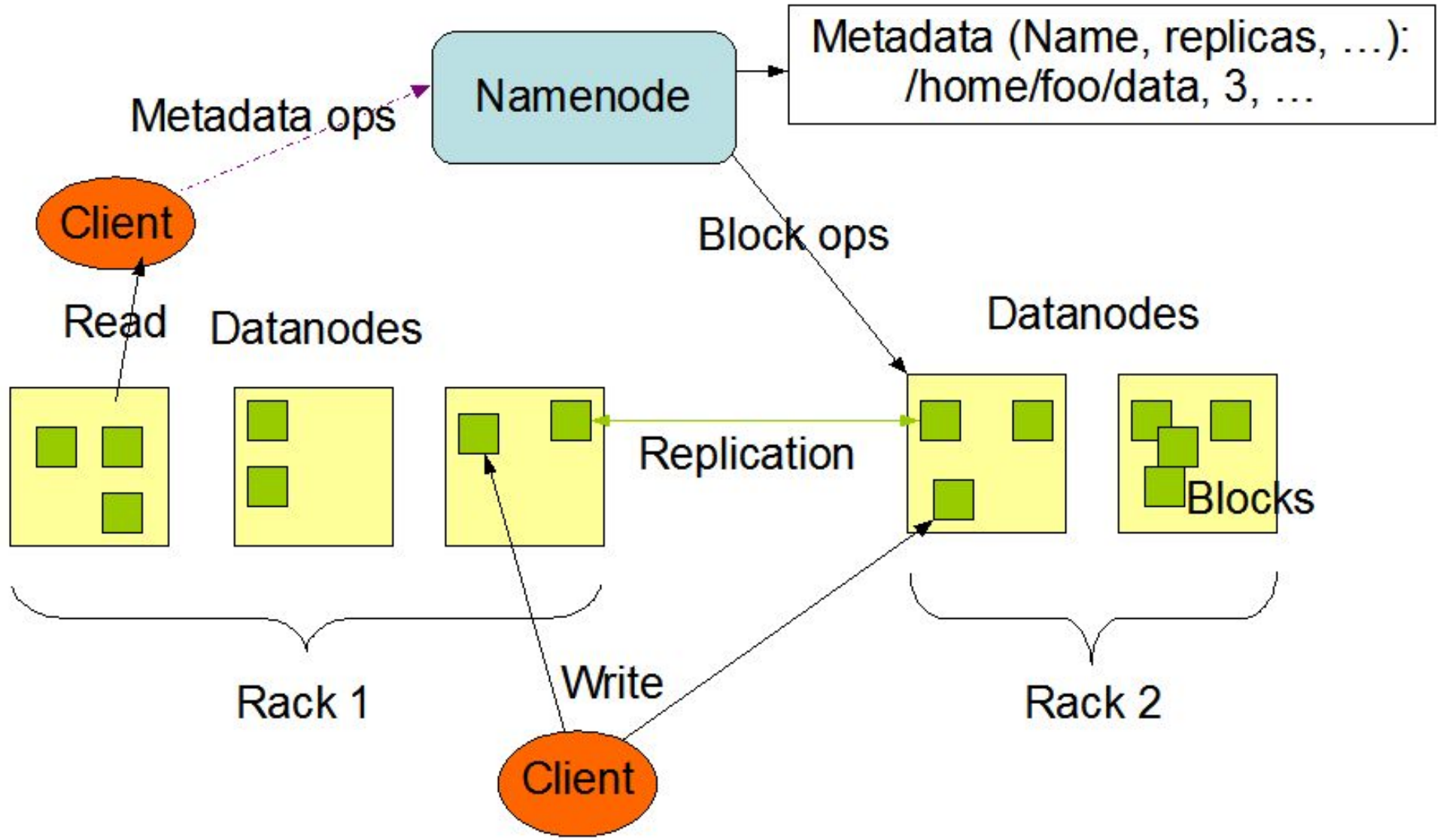
HDFS: Task Tracker

- Other part of slave node.
- Has the actual logic to perform the task.
- Performs the task (Map and reduce functions) on the data assigned to it by Master Node

HDFS continued..

- The NameNode and DataNode are pieces of software designed to run on commodity machines.
- These machines typically run a GNU/Linux operating system (OS).
- HDFS is built using the Java language; any machine that supports Java can run the NameNode or the DataNode software.
- Usage of the highly portable Java language means that HDFS can be deployed on a wide range of machines.
- A typical deployment has a dedicated machine that runs only the NameNode software.
- Each of the other machines in the cluster runs one instance of the DataNode software.

HDFS Architecture



Failure of Master Node

Name Node failure

- Since Name Node contains all the important information required for the data distribution and computation.
- If it crashes, HDFS could fail.
- To overcome this single point failure Hadoop introduced the Backup Node.
- Backup Node regularly contacts Name Node and maintains an up-to date snapshots of Name Node's directory information.
- When Name Node fails Backup helps to revive the Name Node.

Job Tracker failure

- Again, a single point failure.
- If it goes down, all running jobs fail.
- In the latest Hadoop architecture Job Tracker is split into application logic such that there can be a number of masters having different logic.
- Thus, the cluster is capable of supporting different application logic

Rack Awareness in Hadoop HDFS

What is a rack?

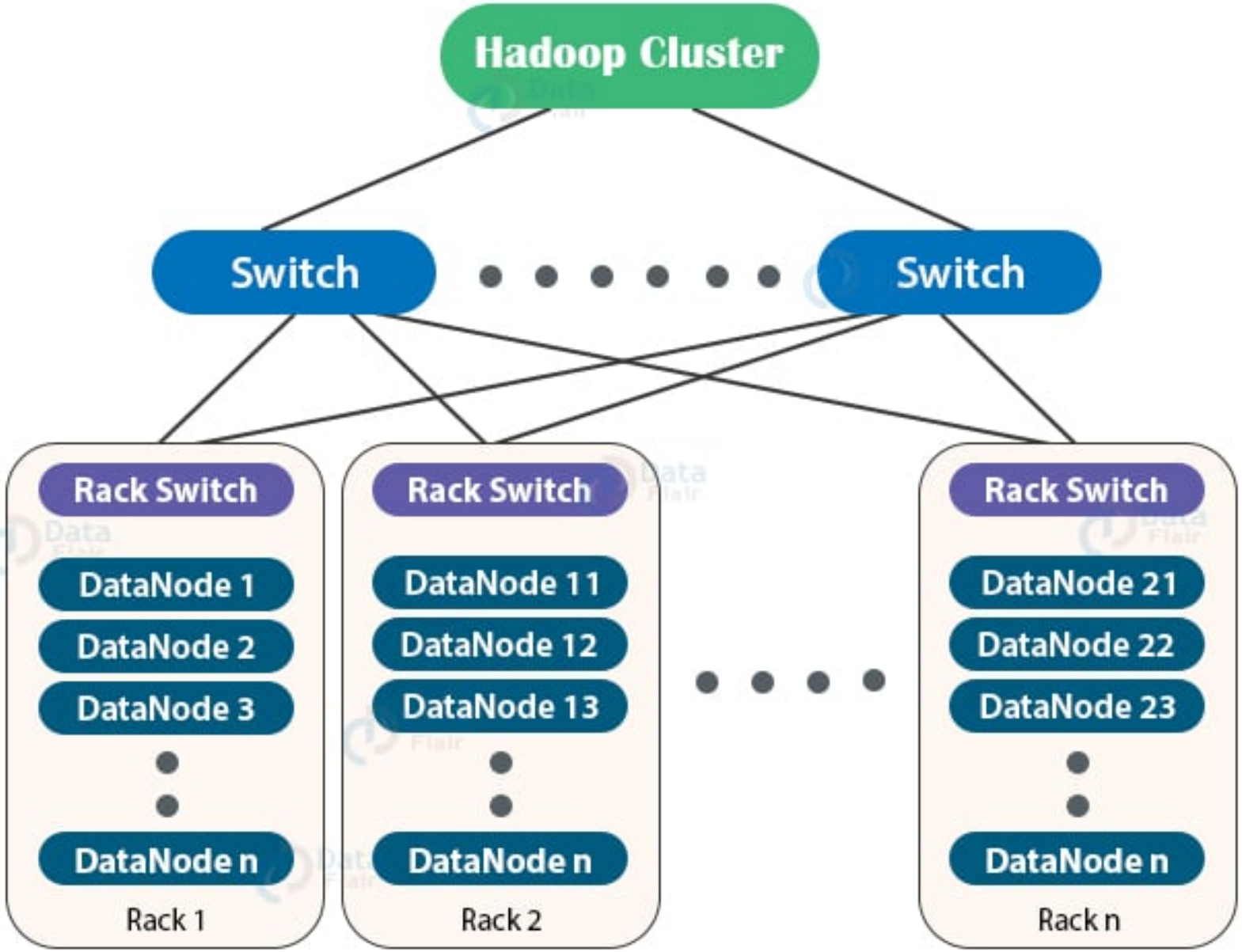
- The **Rack** is the collection of around 40-50 DataNodes connected using the same network switch.
- If the network goes down, the whole rack will be unavailable.
- A large Hadoop cluster is deployed in multiple racks.

Rack awareness

- HDFS stores files across multiple nodes (DataNodes) in a cluster.
- To get the maximum performance from Hadoop and to improve the network traffic during file read/write, NameNode chooses the DataNodes on the same rack or nearby racks for data read/write.
- Rack awareness is the concept of choosing the closer DataNode based on rack information.

Rack awareness continued...

- In a large Hadoop cluster, there are multiple racks.
- Each rack consists of DataNodes.
- Communication between the DataNodes on the same rack is more efficient as compared to the communication between DataNodes residing on different racks.
- To reduce the network traffic during file read/write, NameNode chooses the closest DataNode for serving the client read/write request.
- NameNode maintains **rack ids** of each DataNode to achieve this rack information.
- This concept of choosing the closest DataNode based on the rack information is known as **Rack Awareness**.


Hadoop Cluster
Switch
Switch
Rack Switch
DataNode 1
DataNode 2
DataNode 3
DataNode n
Rack 1
Rack Switch
DataNode 11
DataNode 12
DataNode 13
DataNode n
Rack 2
Rack Switch
DataNode 21
DataNode 22
DataNode 23
DataNode n
Rack n

Replica Placement via Rack Awareness

HDFS stores replicas of data blocks of a file to provide:

1. Fault tolerance
2. High availability

The network bandwidth between nodes within the rack is higher than the network bandwidth between nodes on a different rack.

Replica Placement via Rack Awareness continued…..

Store replicas on different nodes on the same rack:

It improves the network bandwidth, but if the rack fails (rarely happens), then there will be no copy of data on another rack.

Store replicas on unique racks:

Due to the transfer of blocks to multiple racks while writes increase the cost of writes.

Replica Placement via Rack Awareness continued…..

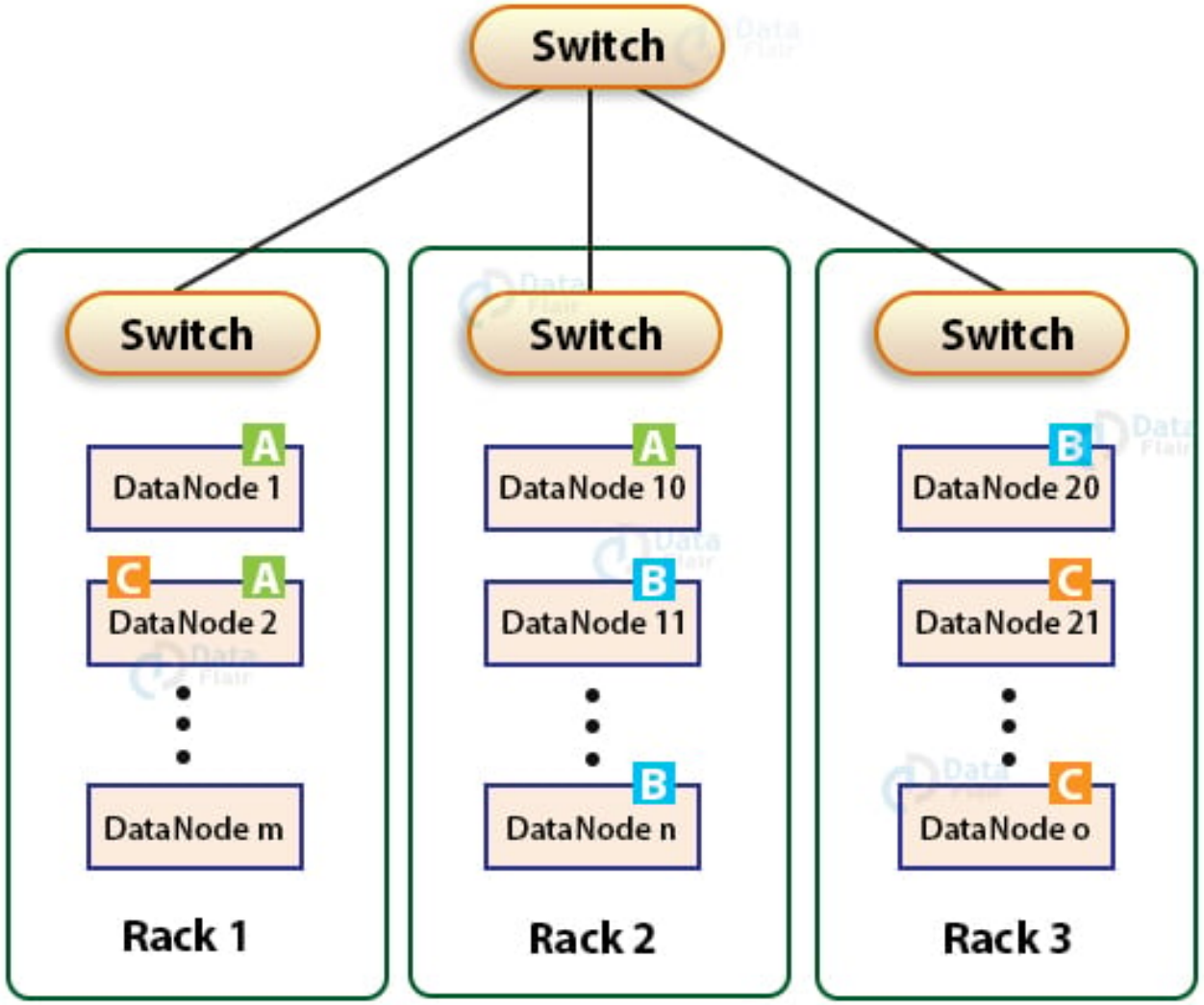
Rack awareness policies:

- Not more than one replica be placed on one node.
- Not more than two replicas are placed on the same rack.
- Also, the number of racks used for block replication should always be smaller than the number of replicas.

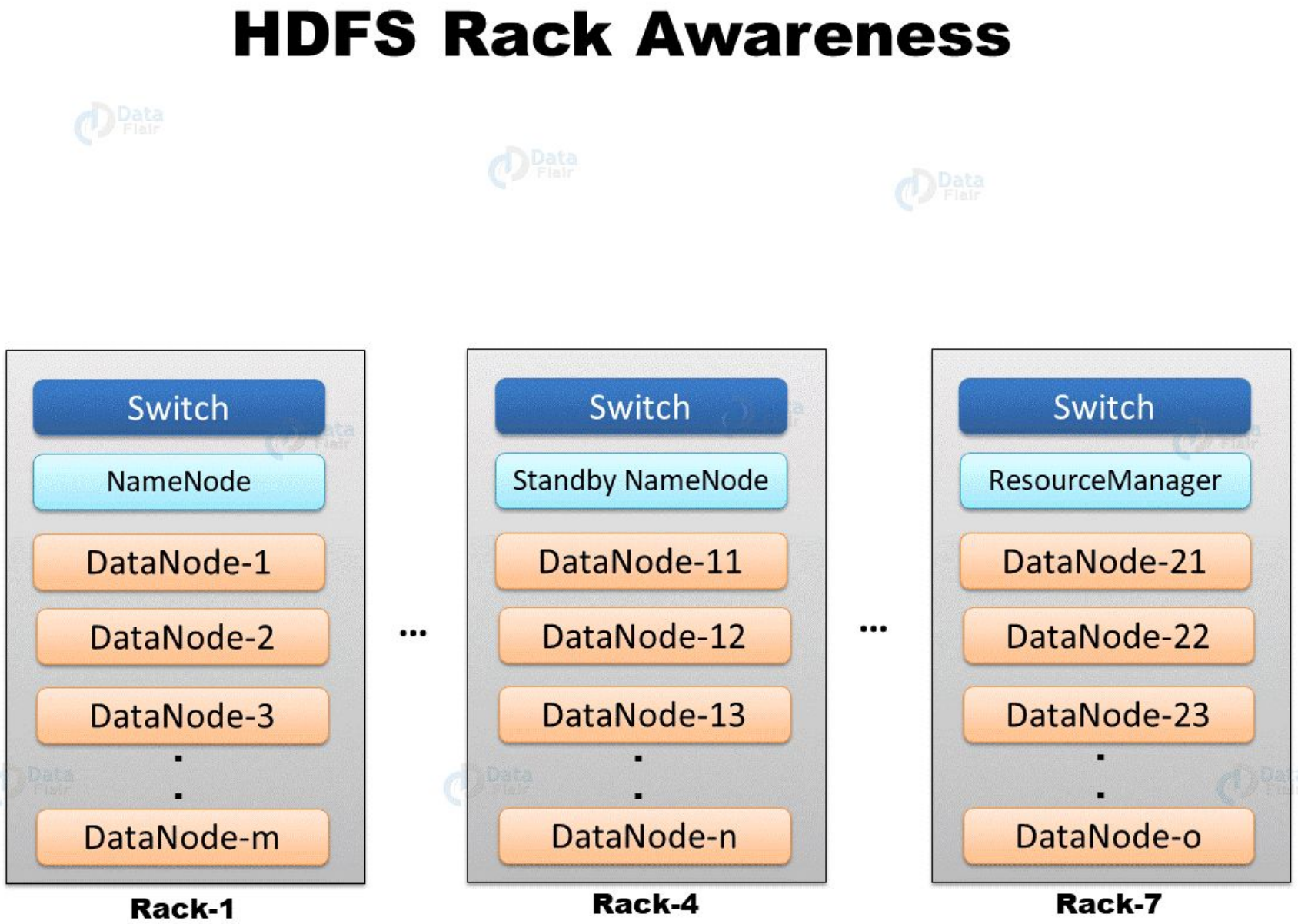
This policy improves write performance and network traffic without compromising fault tolerance.

Replica Placement via Rack Awareness continued…..

- A file “File.txt” divided into three blocks A, B, and C.
- HDFS creates replicas of blocks
- Places the first copy of each block on the closest DataNode
- The second replica of each block on different DataNode on the same rack.
- The third replica on different DataNode on a different rack.



Replica Placement via Rack Awareness continued



GIF : Rack Awareness

Drawbacks/ limitations

- Since Name Node is the single point for storage and management of metadata, this can be a bottleneck for supporting a huge number of files, especially a large number of small files.
- For a very robust system replica distribution should be on totally separate racks.
- But would lead to high communication overhead especially during write operation, as all replicas must be updated.
- Thus HDFS write two of the replicas on different nodes of the same rack and write the other one on a totally separate rack.

Failure of Slave Node

Data Node Failure

- Data Node send a periodic heartbeat to Name Node.
- Missing heartbeat helps detected Data Node failure.
- In such a case Name Node removes the crashed Data Node from the cluster and redistributes its chunks to other surviving Data Nodes.

Task Tracker Failure

- Just like Data Node, Task Tracker's failure is detected by the Job Tracker after missing its heartbeat.
- Job Tracker then decides the next step:
  - It may reschedule the job elsewhere. OR
  - It may mark specific record to avoid . OR
  - It may blacklist this Task Tracker as unreliable.

References

- https://explodingtopics.com/blog/data-generated-per-day?utm_source=chatgpt.com
- https://hadoop.apache.org/docs/r1.2.1/index.html
- https://data-flair.training/blogs/rack-awareness-hadoop-hdfs/#:~:text=Therefore%2C%20NameNode%20on%20multiple%20rack,placed%20on%20the%20same%20rack