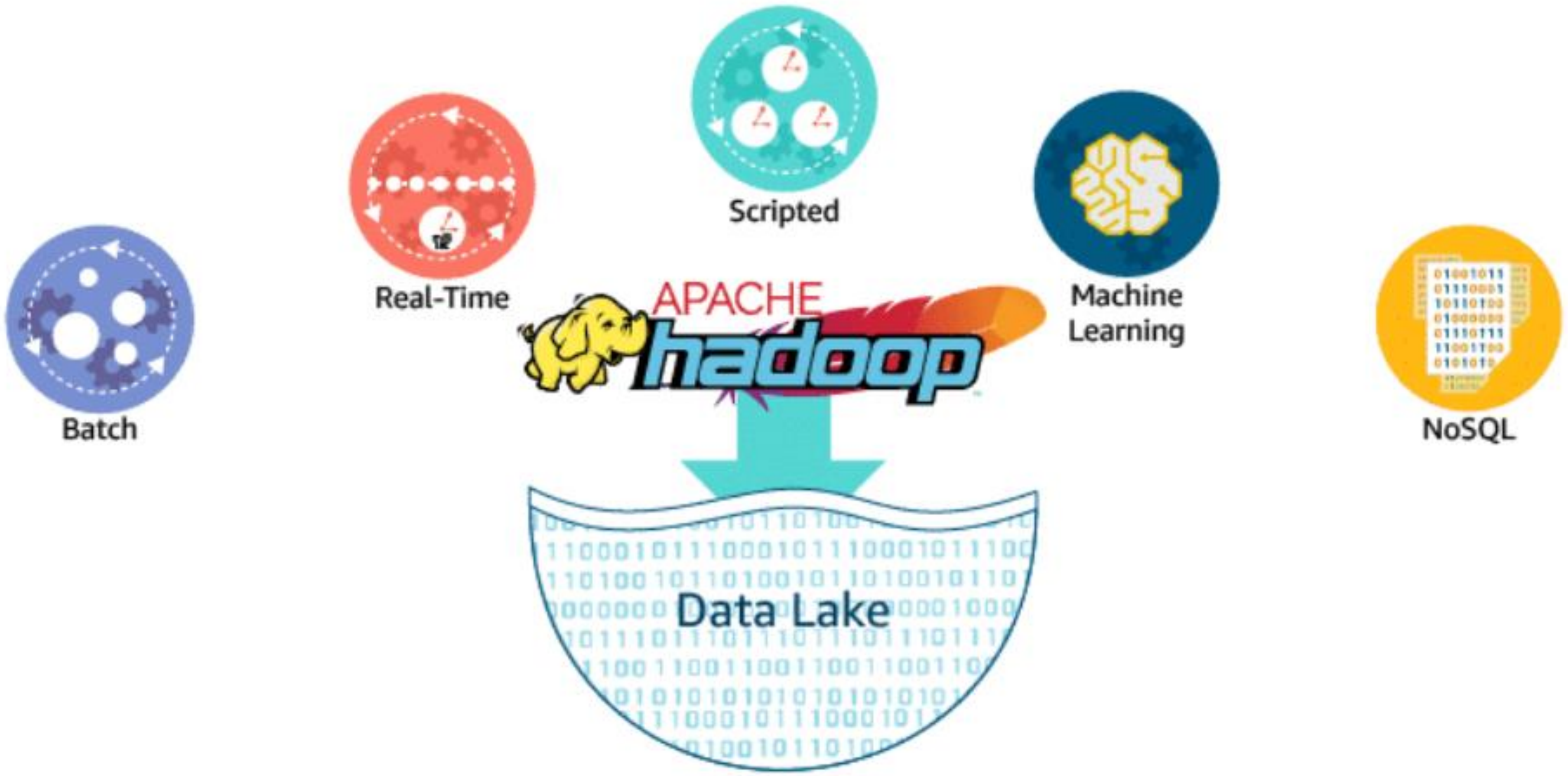
Basics of Hadoop

Apache Hadoop

- Hadoop uses a distributed processing architecture, in which a task is mapped to a cluster of commodity servers for processing.
- Each piece of work distributed to the cluster servers can be run or re-run on any of the servers.
- The cluster servers frequently use the Hadoop Distributed File System (HDFS) to store data locally for processing.
- The results of the computation performed by those servers are then reduced to a single output set.
- One node, designated as the master node, controls the distribution of tasks and can automatically handle server failures.


Scripted
Real-Time
Machine Learning
Batch
NoSQL
APACHE hadoop
Data Lake

Benefits of using Apache Hadoop

- Unlike traditional database systems, Hadoop can process structured, semistructured, or unstructured data. This includes virtually any data format currently available.
- In addition to natively handling many types of data (such as XML, CSV, text, log files, objects, SQL, JSON, and binary), you can use Haddop to transform data into formats that allow better integration into your existing data sets.
- Also, you can store data with or without a schema and perform large-scale ETL operations to transform your data.

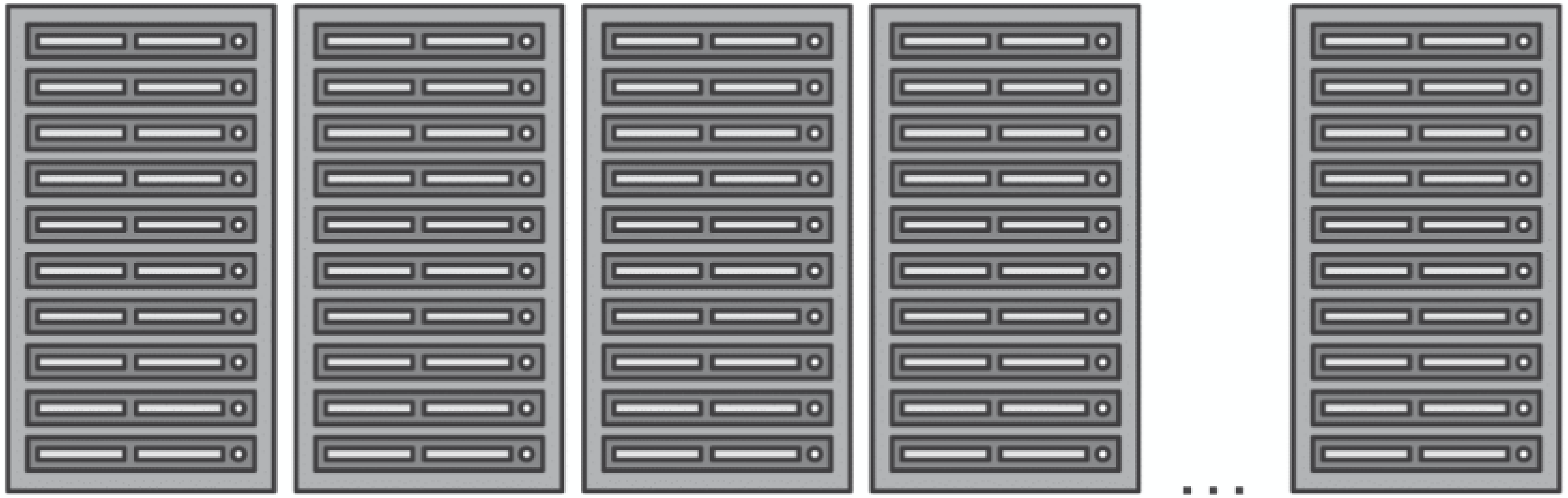
Hadoop can handle variety of data

CSV
XML
JSON
Amazon RDS
Amazon Aurora
Amazon DynamoDB
Amazon ElastiCache
MySQL
MariaDB
Amazon Neptune
PostgreSQL
Amazon S3
Amazon Redshift Spectrum
SQL Server
ORACLE

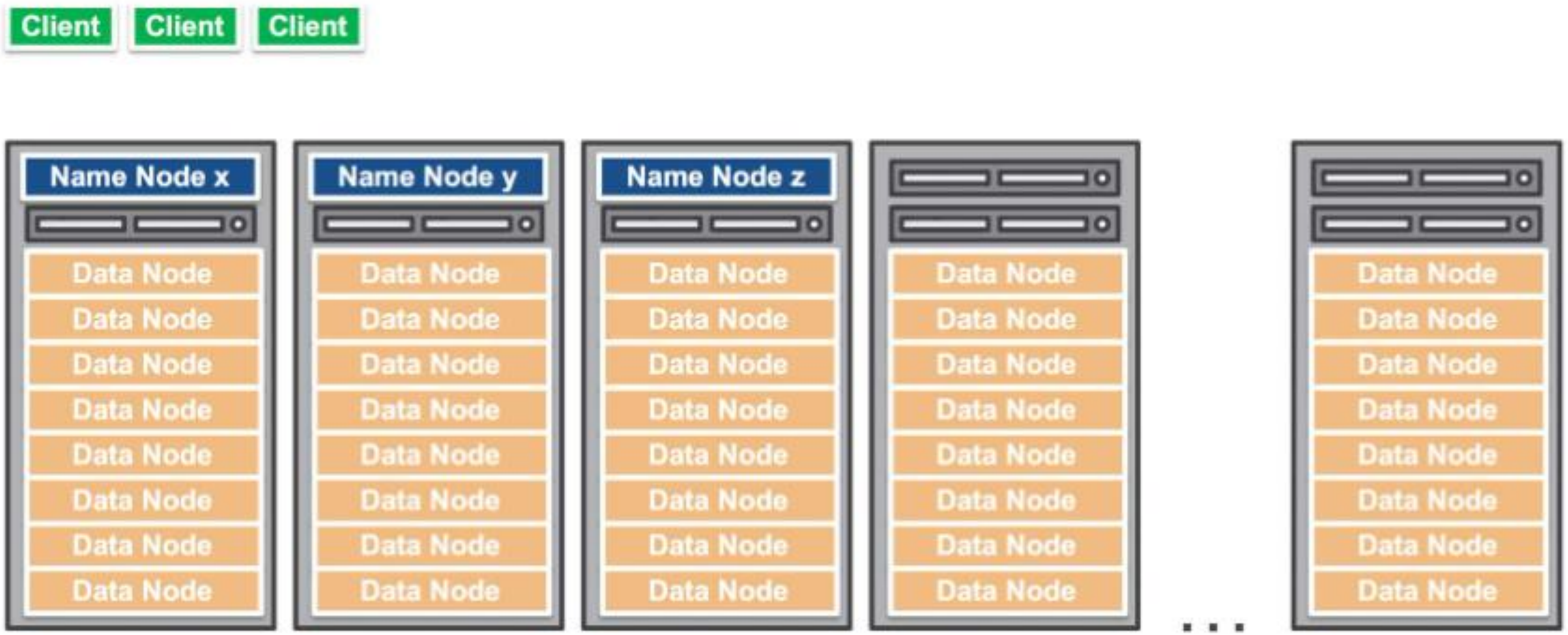
HDFS

- To handle massive volumes of data rapidly, the processing system required a way to distribute the load of reading and writing files across tens or even hundreds of high-powered servers.
- HDFS is distributed storage allowing files to be read and written to clusters of servers in parallel. This dramatically reduces the overall length of each and every operation.
- An HDFS cluster primarily consists of a NameNode, which manages the file system metadata, and DataNodes, which store the actual data.

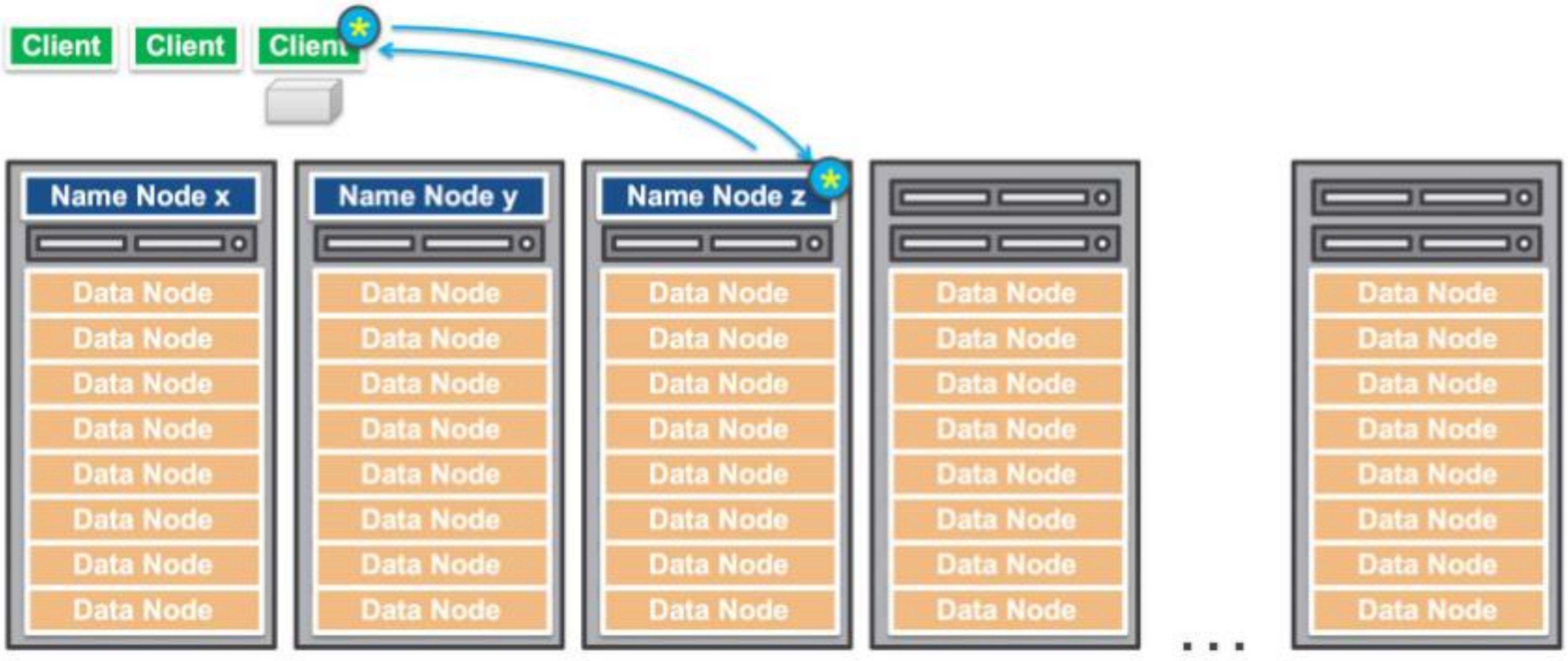
HDFS cont..



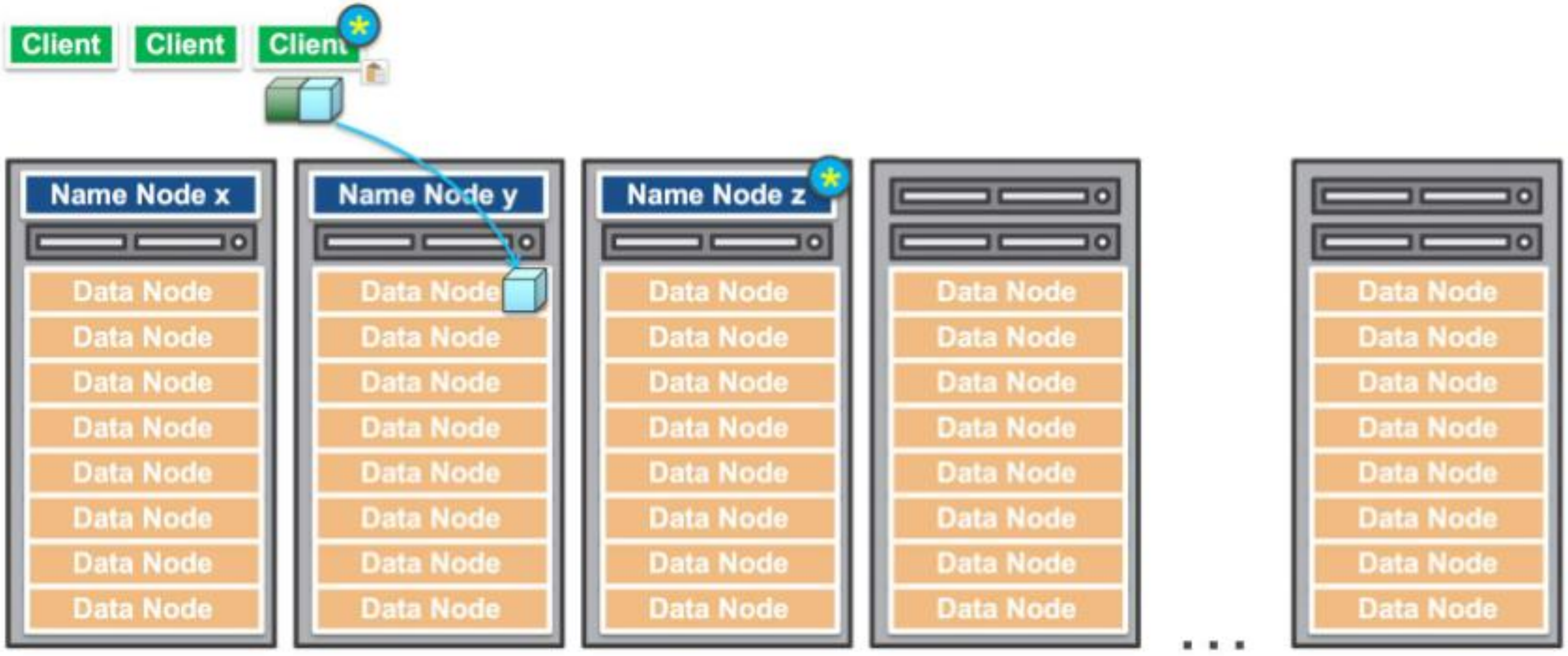
- Let’s start by taking a look at the basic layout of a Hadoop cluster, focusing on HDFS. You start with a large number of dedicated commodity servers or virtual servers.



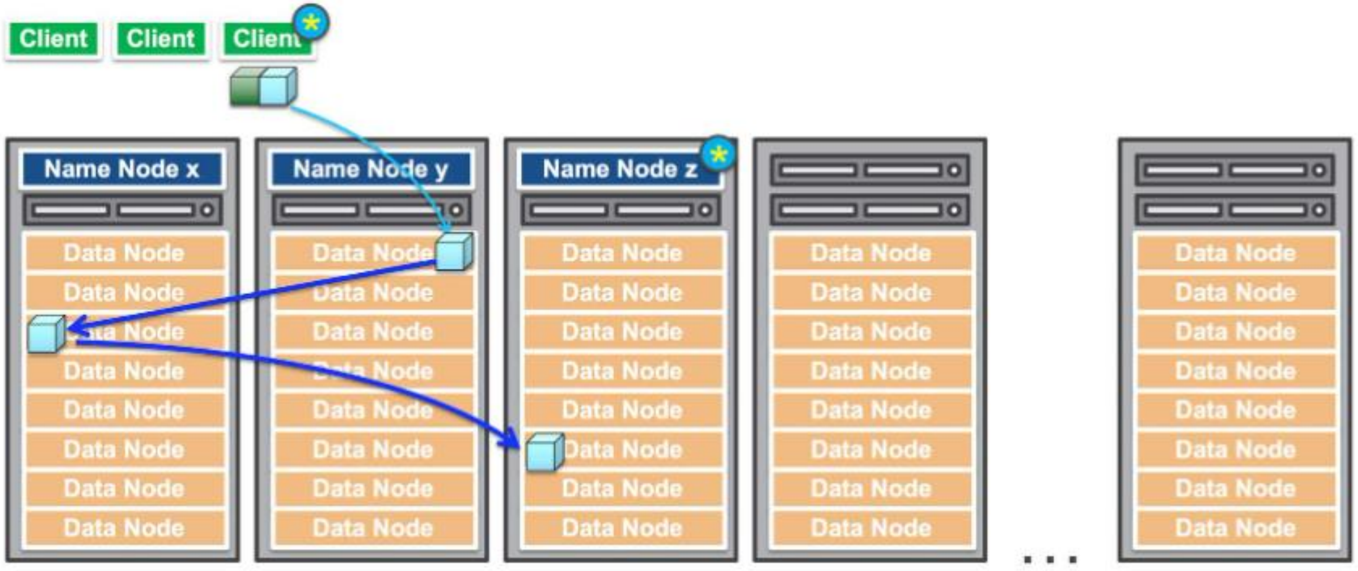
- We have multiple DataNodes, where the actual data will be stored, and a small number of servers as NameNodes, which contain metadata of where the data resides.
- Clients contact a NameNode for file metadata or file modifications and perform actual file I/O directly with the DataNodes.



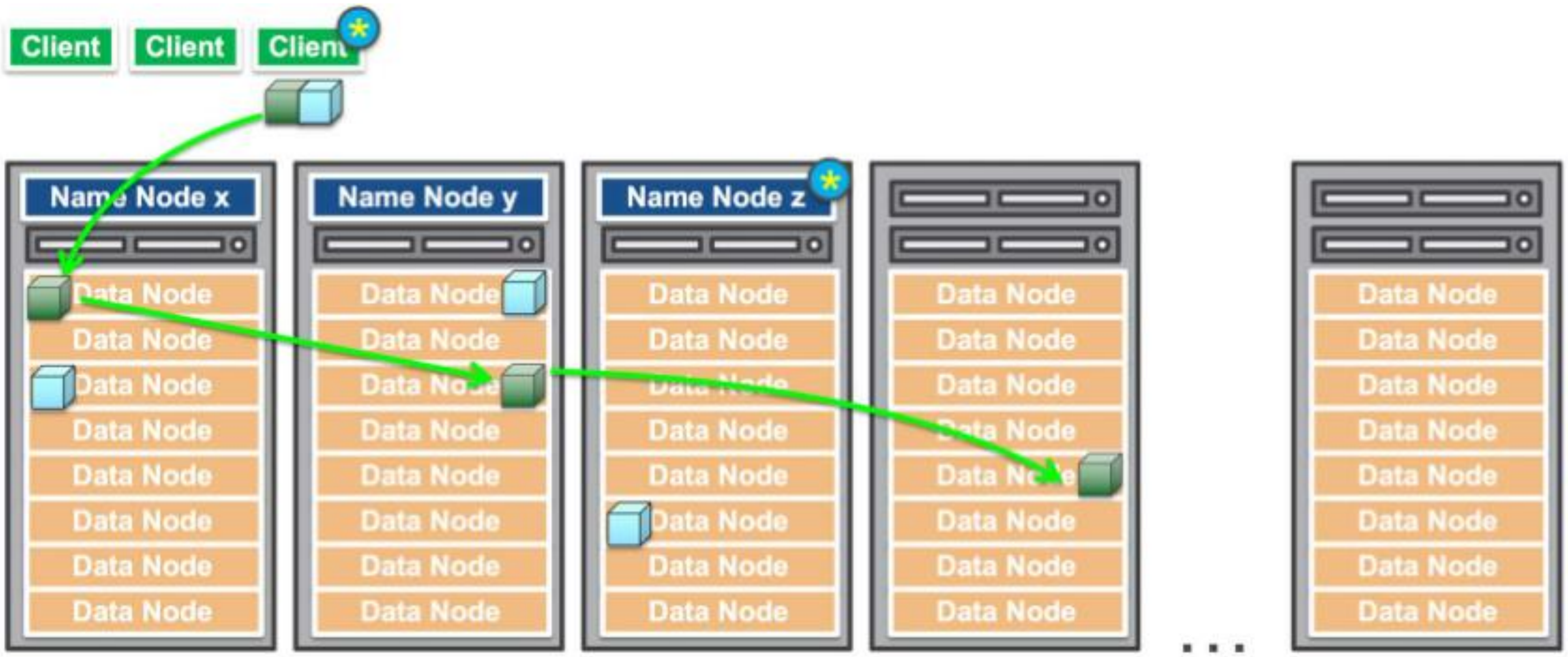
- In this example, everything is configured, so let’s take a look at a write to this HDFS cluster. A Client issues a write to a NameNode.
- The NameNode performs various checks to make sure that the file doesn’t already exist and that the client has the permissions to create the file.
- The NameNode returns the location of where to write each data block that constitutes the request.



- The Client then writes the data block to the assigned DataNode.



- In this example, we have several racks configured and have let the name nodes know which data nodes are on which racks.
- As soon as the Client finishes writing the data block, the DataNode starts copying the data block to another DataNode. Hadoop is configured to store three copies of every data block by default. Therefore, the second DataNode copies the data block to a node on a different rack.



- This process is repeated for all of the blocks that constitute the write request.
- The original Client only writes each block once to the cluster, which internally handles the additional two writes per block.
- The NameNode maintains a mapping of where all the data blocks reside.