Figure 1-1 illustrates the “classic” Hadoop architecture for batchmode

analytics and data warehousing, focusing on the aspects that

are important for our discussion.



*Figure 1-1. Classic Hadoop architecture*

In this figure, logical subsystem boundaries are indicated by dashed

rectangles. They are clusters that span physical machines, although

HDFS and YARN (Yet Another Resource Negotiator) services share

the same machines to benefit from data locality when jobs run.

Functional areas, such as persistence, are indicated by the rounded

dotted rectangles.

Data is ingested into the persistence tier, into one or more of the following:

HDFS (Hadoop Distributed File System), AWS S3, SQL and

NoSQL databases, and search engines like Elasticsearch. Usually this is done using special-purpose services such as Flume for log aggregation

and Sqoop for interoperating with databases.

Later, analysis jobs written in Hadoop MapReduce, Spark, or other

tools are submitted to the Resource Manager for YARN, which

decomposes each job into tasks that are run on the worker nodes,

managed by Node Managers. Even for interactive tools like Hive

and Spark SQL, the same job submission process is used when the

actual queries are executed as jobs.

Table 1-1 gives an idea of the capabilities of such batch-mode

systems.

*Table 1-1. Batch-mode systems*

|  |  |
| --- | --- |
| **Metric** | **Sizes and units** |
| Data sizes per job | TB to PB |
| Time between data arrival and processing | Many minutes to hours |
| Job execution times | Minutes to hours |

So, the newly arrived data waits in the persistence tier until the next

batch job starts to process it.