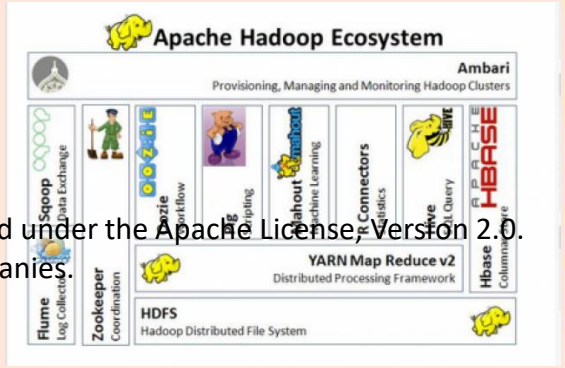
CBTU presents a course on **Big data and Hadoop**

Module 2: hadoop

Section 2.1: Hadoop architecture – assumptions and goals



BIG DATA

Assumptions and Goals

- Hardware Failure / Quick recovery
- Batch processing than real-time
- Streaming Data Access
- Simple Coherency Model
- HDFS is designed to work with large datasets
- Moving Computation is Cheaper than Moving Data

Hardware Failure

- A HDFS instance may consist of hundreds or thousands of server machines, each storing part of the file system's data.
- Detection of faults and quick, automatic recovery from them is a core architectural goal of HDFS.

Batch processing than real-time

- HDFS is designed more for batch processing rather than interactive use by users.
- The emphasis is on high throughput of data access rather than low latency of data access.

Streaming Data Access

- Applications that run on HDFS need streaming access to their data sets.

Simple Coherency (consistency) Model

- HDFS applications need a write-once-read-many access model for files.
- This assumption simplifies data coherency issues and enables high throughput data access.
- A MapReduce application or a web crawler application fits perfectly with this model.

HDFS - Large Data sets

- HDFS support large files of GB, TB, PB in size.
- Provides high aggregate data bandwidth and scale to hundreds of nodes in a single cluster.
- Supports tens of millions of files in a single instance.

Moving Computation is Cheaper than Moving Data

- When dataset is large, a computation requested by an application is much more efficient if it is executed near the data it operates on.
- This minimizes network congestion and increases the overall throughput of the system.

Portability

Portability Across Heterogeneous Hardware and Software Platforms.

- HDFS has been designed to be easily portable from one platform to another. This facilitates widespread adoption of HDFS as a platform of choice for a large set of applications.

Robustness

- The primary objective of HDFS is to store data reliably even in the presence of failures. The three common types of failures are NameNode failures, DataNode failures and network partitions.

Thanks for watching

CBTUniversity.com
learnq@CBTUniversity.com
+91 963 246 5599
/CBTUniversity