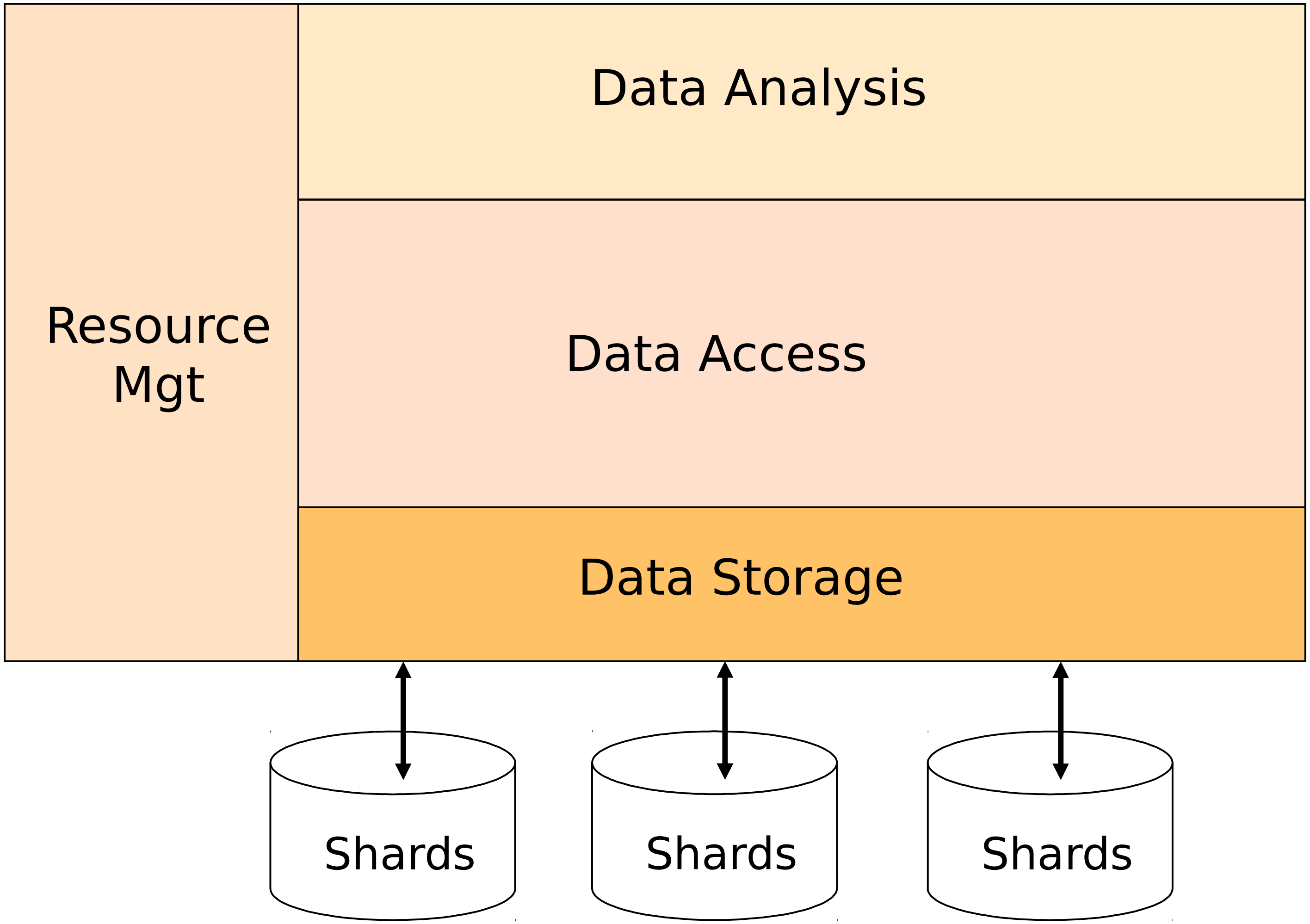
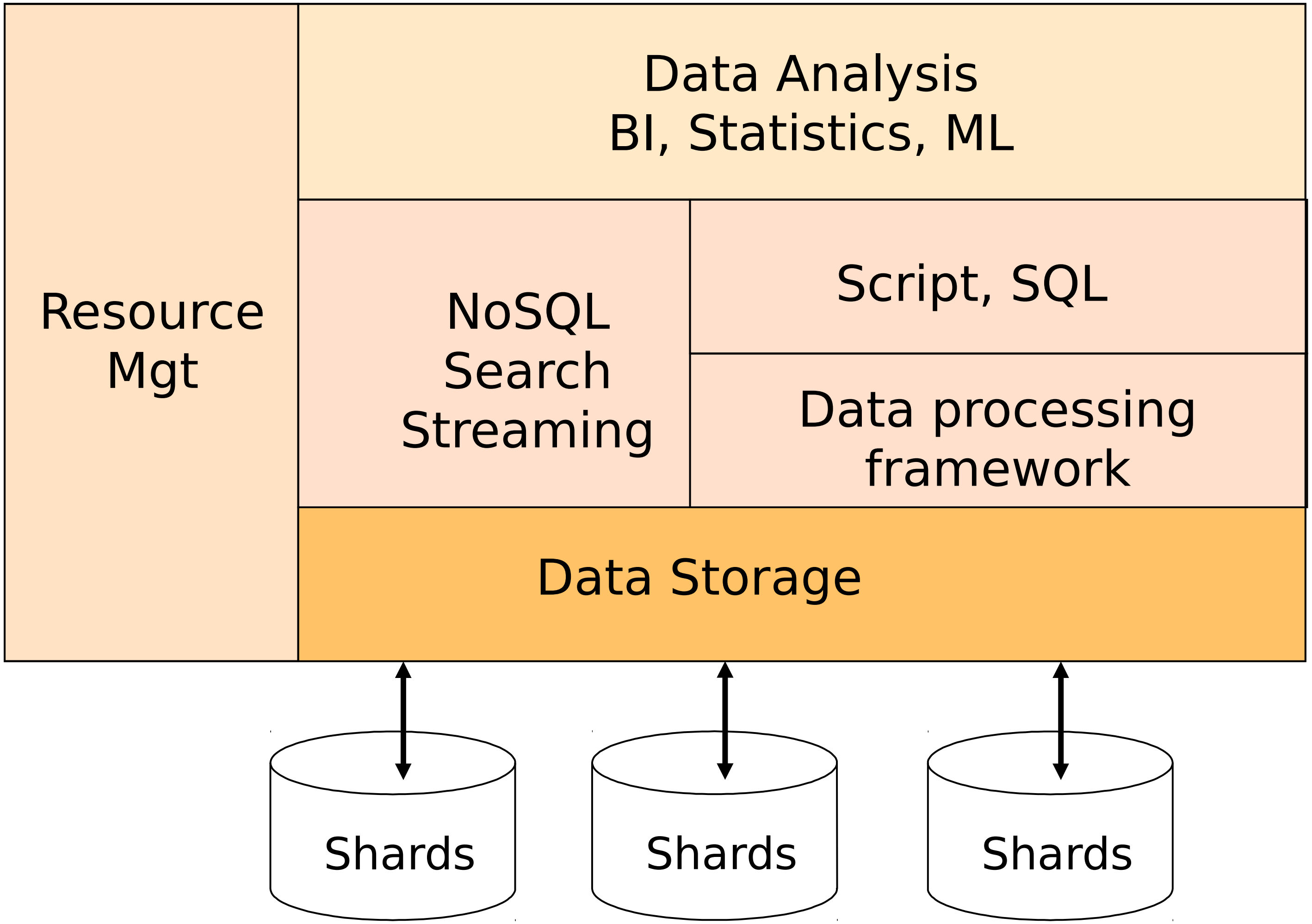
Big Data Architectures

1. The big data software stack
2. Apache Hadoop

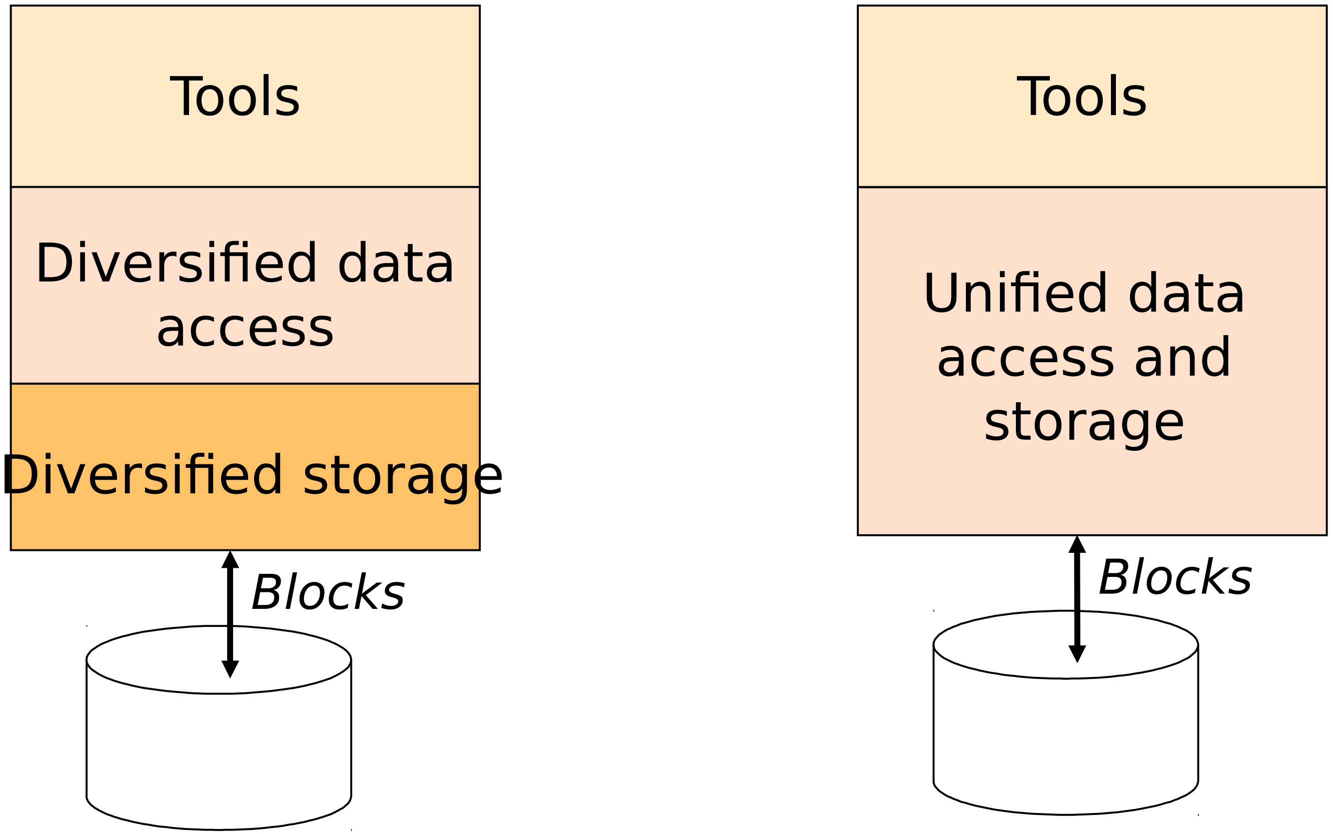
1. Big Data Software Stack



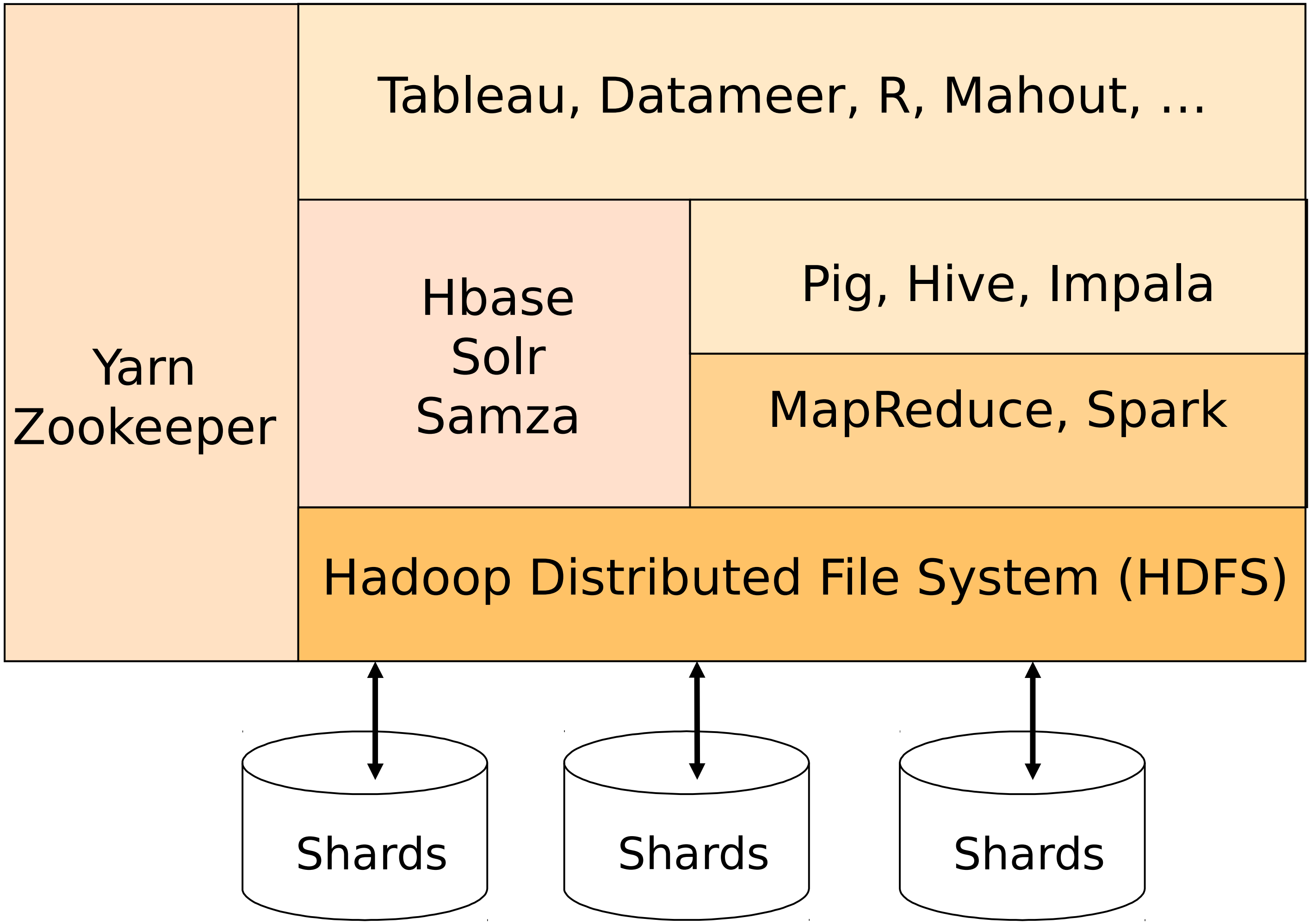
Big Data Software Stack



Comparison with RDBMS



2. Apache Hadoop Architecture

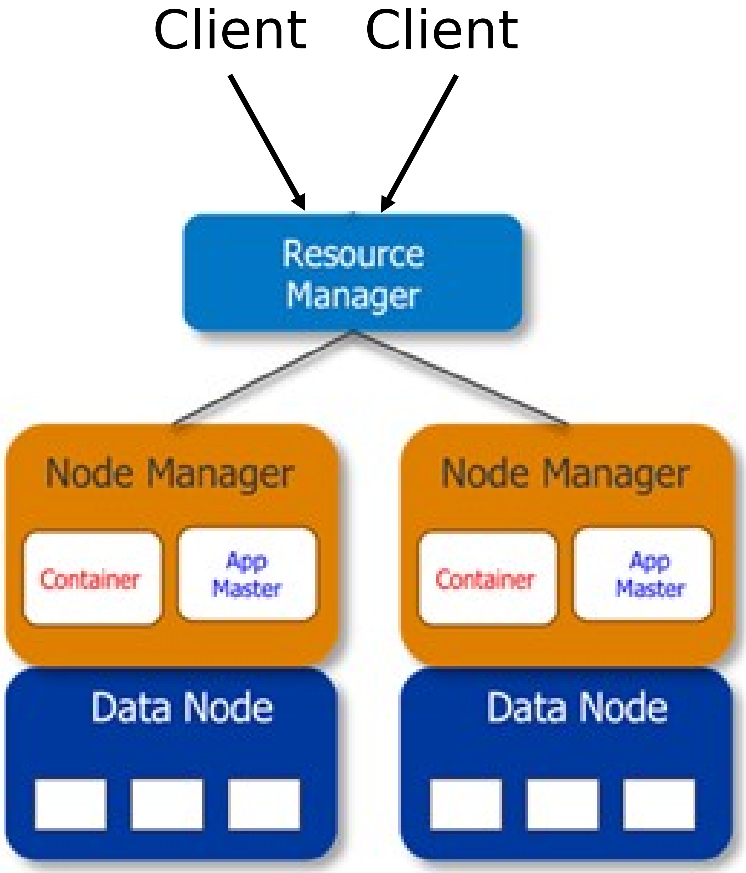


Cluster Management with Yarn

- Yarn (Yet another Resource Negociator)
  - Originally strongly coupled with HDFS and MapReduce for batch processing
- Hadoop 3.0: a distributed OS for all types of big data applications (interactive, streaming, …)
  - Centralized management of system resources between applications
  - Distributed monitoring of execution on each node of the cluster
  - Possibility of federations (sub-clusters) with separate managers
  - GPU support

Yarn Architecture

- Resource Mgr
  - Scheduler: allocates resources to applications based on contraints
  - AppMgr: submits execution requests to App Masters
- Node Mgr
  - Resource (CPU, RAM) container
  - App Master (1 per app): requests resources to Scheduler



Coordination with Zookeeper

- High-performance coordination service for building distributed applications
  - Naming, e. g. DNS
  - Configuration management
  - Group management and communication
  - DLM and distributed synchronization
- Service replicated on a set of nodes
  - A leader propagates updates on copies
  - Other servers are in read mode
  - Fault-tolerance and failover

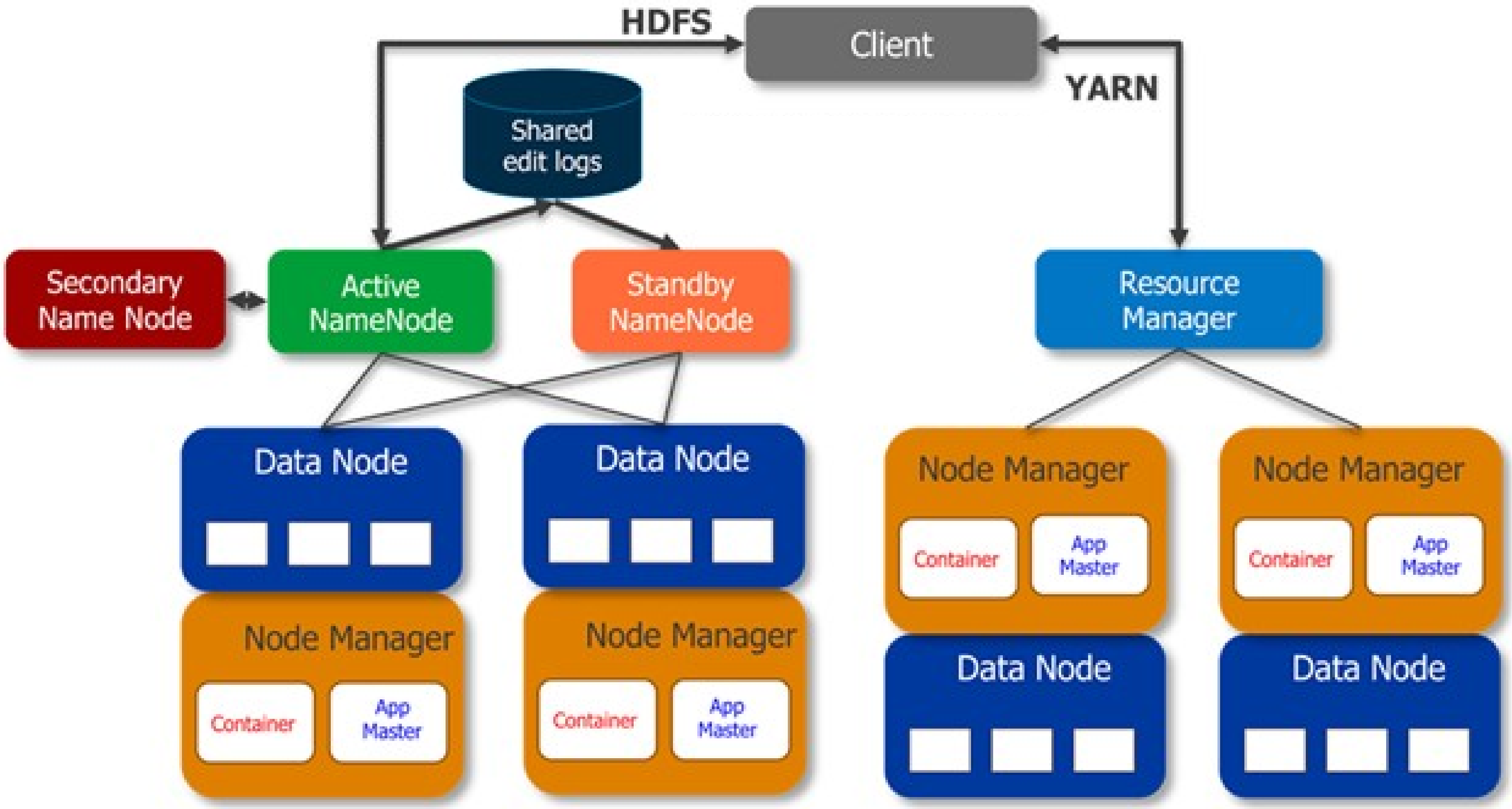
Hadoop Distributed File System (HDFS)

- Origin: Google File System
  - File management for Search Engine, Bigtable, MapReduce, etc.
- Objectives
  - Very large files (terabytes), containing very many elements, e. g. web pages
  - SN cluster with thousands of cheap nodes
    - Node failure is the norm!
  - Performance, fault tolerance and high availability
  - Replication and failover

Types of Nodes

- Name Node
  - Active (Master)
    - Metadata: file conversion, directory, operation log
    - Status of data nodes
  - Secondary: to perform log checkpointing
  - Standby: backup node in case of Master node failure
- Data Node
  - Access to the node's data
    - Linux local file system
  - Replication

HDFS and Yarn



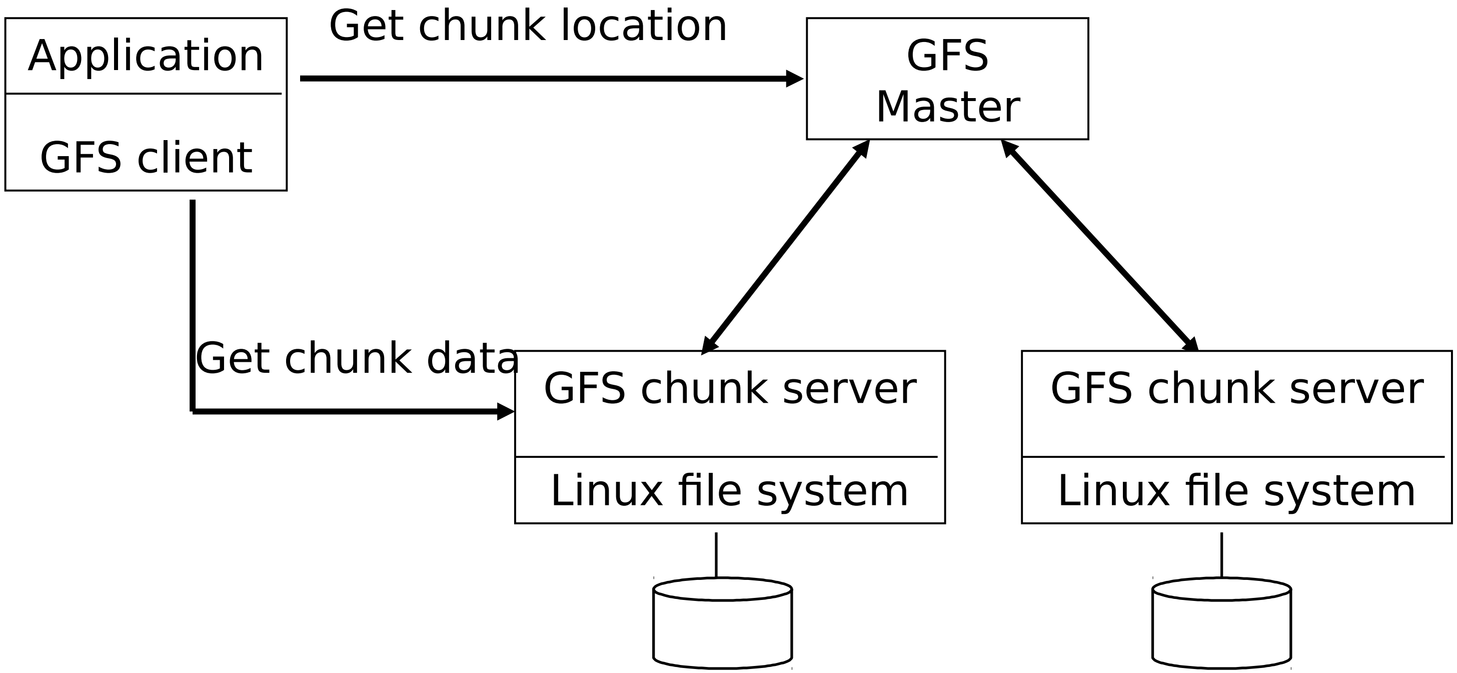
HDFS: design choices

- Files are divided in fixed-size partitions, called *chunks*, of large size, i.e. 64 MB
  - Partitions are distributed across multiple nodes
  - Each partition is replicated at several nodes (3 by default)
- Optimized for read and append
  - Random updates are rare
  - Large reads of bulk data (e.g. 1 MB) and small random reads (e.g. 1 KB)
  - Append operations are also large and there may be many concurrent clients that append the same file
  - High throughput (for bulk data) more important than low latency

HDFS: capabilities

- Traditional file system interface (create, open, read, write, close, and delete file)
  - Two additional operations: snapshot and record append
  - No update: need to do read and write
- Relaxed consistency, with atomic record append
  - No need for distributed lock management
  - Up to the application to use techniques such as checkpointing and writing self-validating records
- Name node and data nodes (Yarn)
  - Replication, fault-tolerance and failover

HDFS Overview


Application
GFS client
Get chunk location
GFS
Master
Get chunk data
GFS chunk server
Linux file system
GFS chunk server
Linux file system