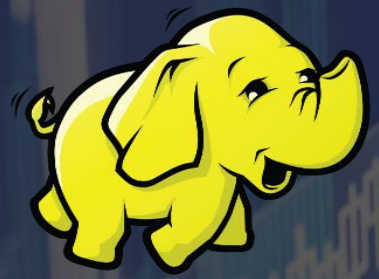
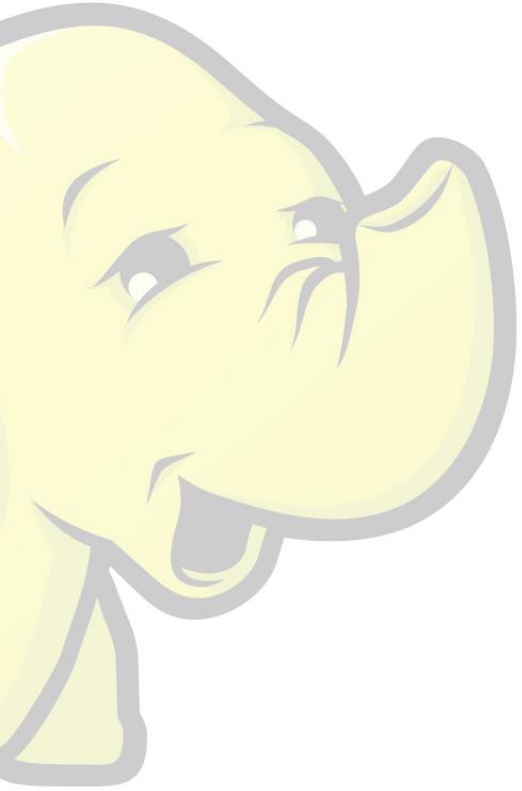
ACADGILD

LEARN. DO. EARN

BIG DATA

DEVELOPMENT





Session 3

HDFS

Agenda

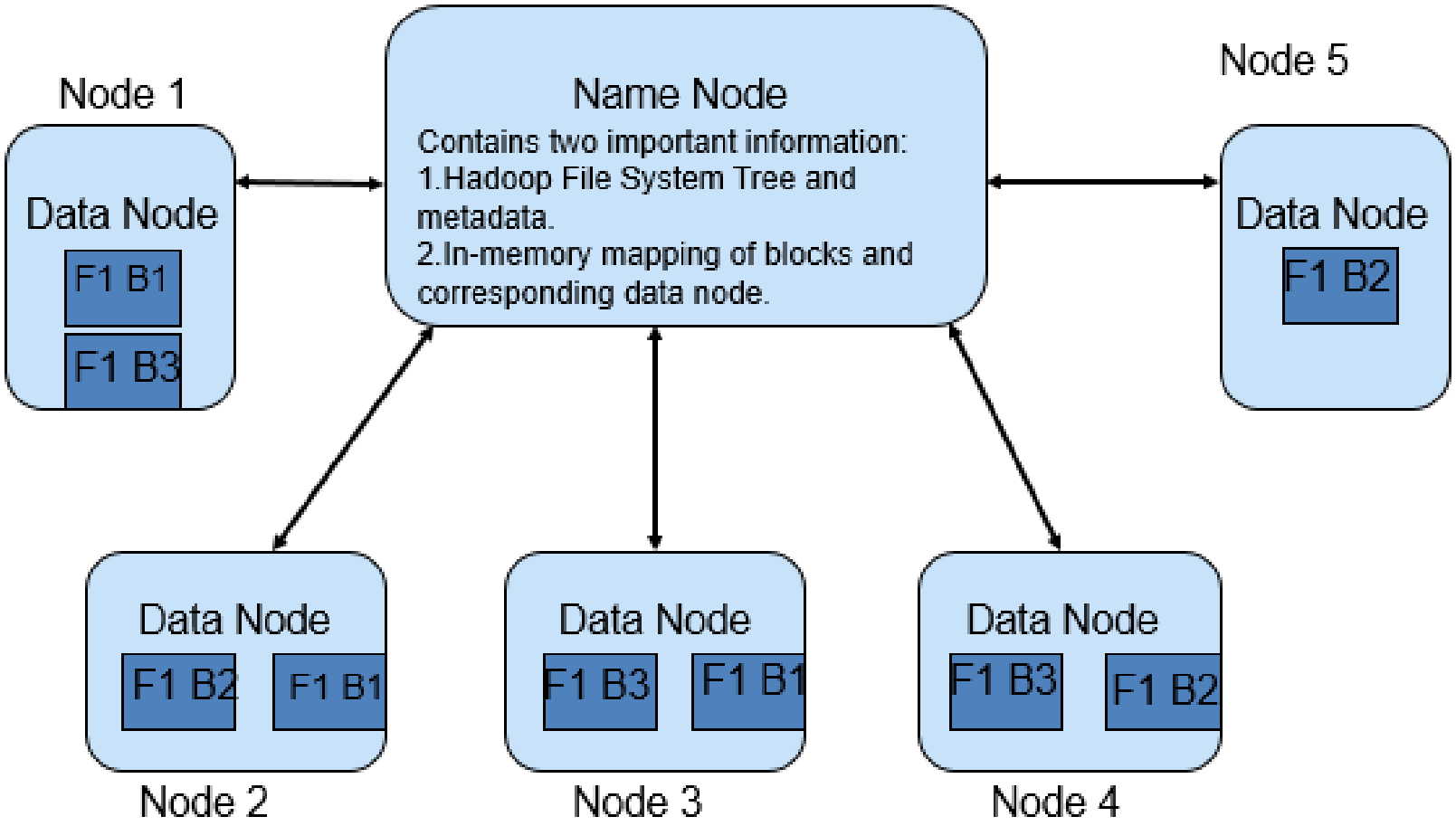
ACADGILD

S. No.	Agenda Title
1.	NameNode and Hadoop Cluster
2.	Arrangement of Racks
3.	Arrangement of Machines and Racks
4.	Local FS and HDFS
5.	NameNode
6.	Checkpointing
7.	Replica Placement
8.	Benefits of Replica Placement and Rack Awareness
9.	URI, URN, and URL
10.	HDFS Commands

S. No.	Agenda Title
11.	Problems with HDFS in Hadoop 1.x
12.	HDFS Federation (Included in Hadoop 2.x)
13.	HDFS Federation
14.	High Availability
15.	Configuration Files in Hadoop
16.	HDFS Configurations

NameNode and Hadoop Cluster

ACADGILD



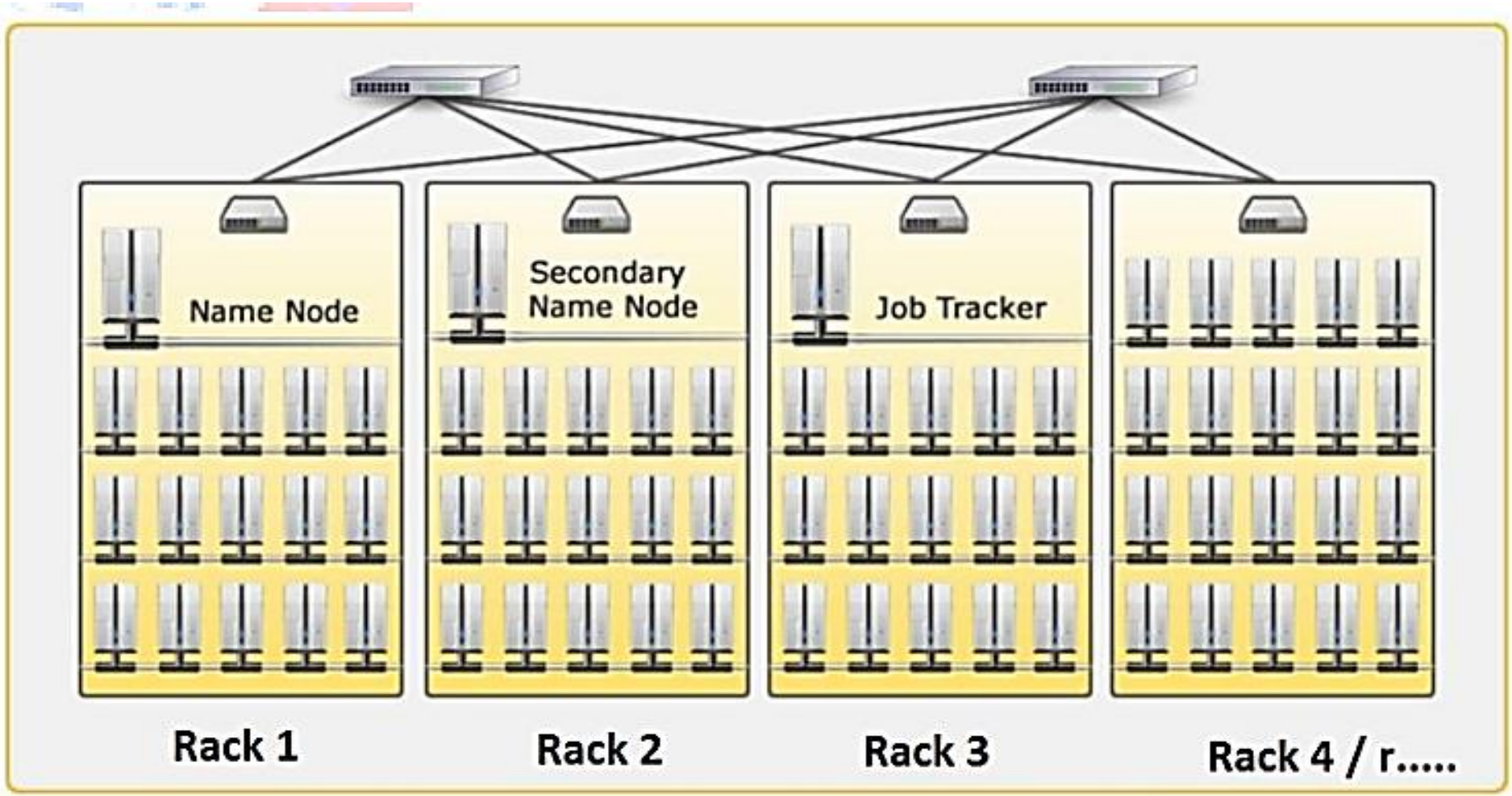
Hadoop Cluster with large number of connected machines, called nodes.

Arrangement of Racks

ACADGILD

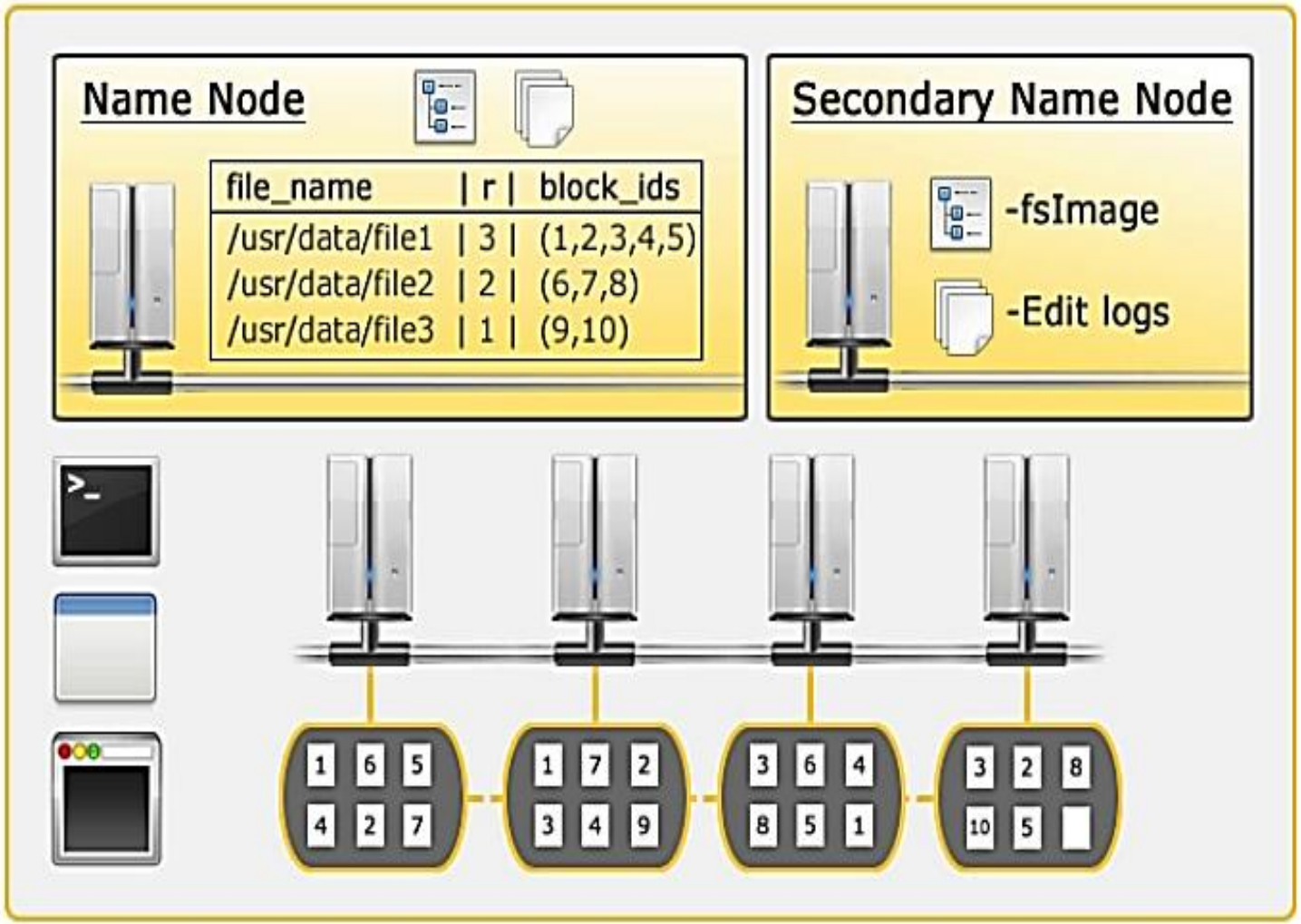
Arrangement of Machines and Racks

ACADGILD



Arrangement of Machines and Racks (Contd.)

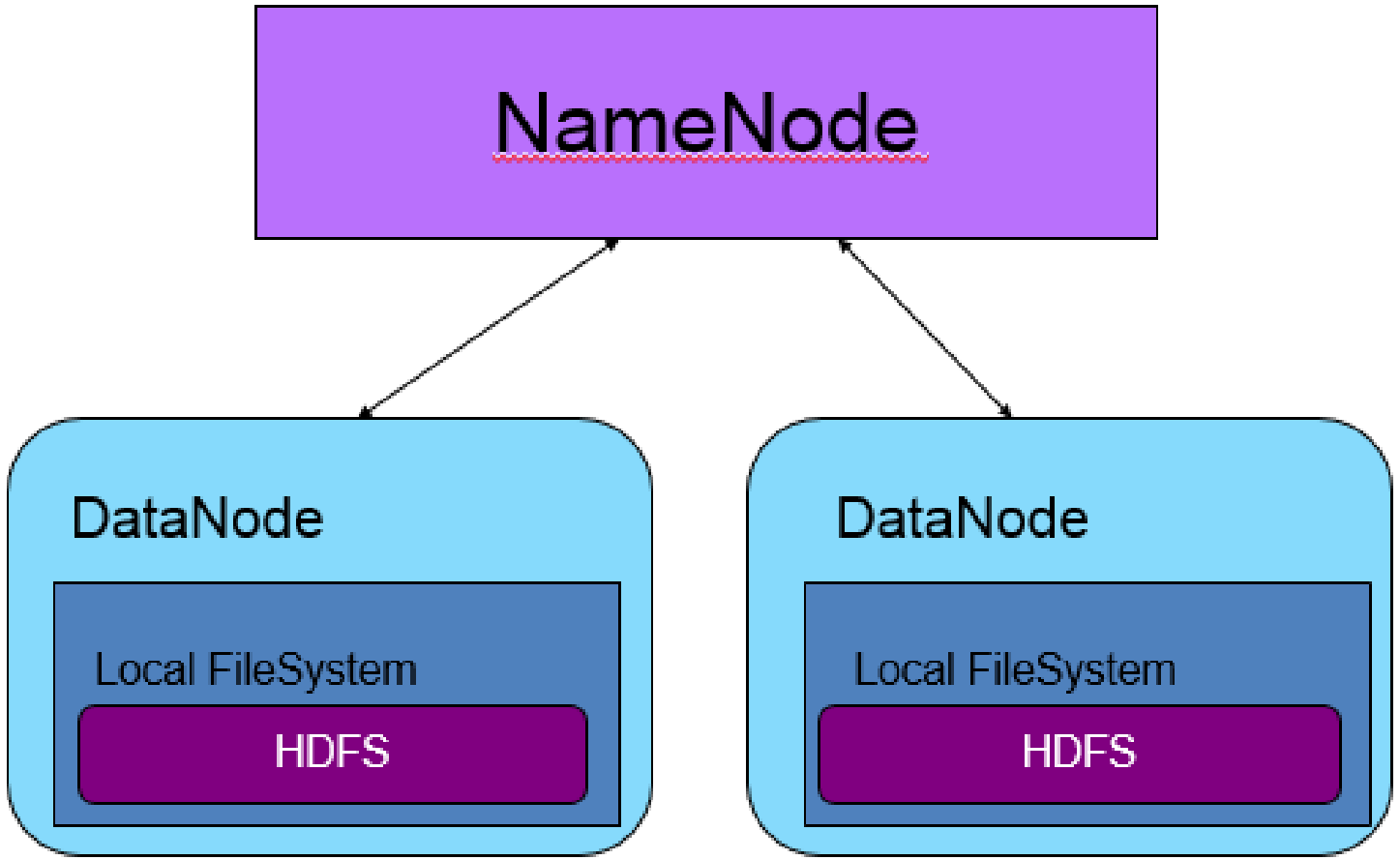
ACADGILD



Local FS and HDFS

ACADGILD

- Some Portion of the Local FS of every participating machine is used in the HDFS.



NameNode

ACADGILD

- A NameNode contains two important files on its hard disk:
  1. **fsimage (file system image)** contains:
     - A directory structure of the HDFS
     - Replication level of the files
     - Modification and access times of files
     - Access permissions of files and directories
     - Block size of files
     - Blocks constituting a file

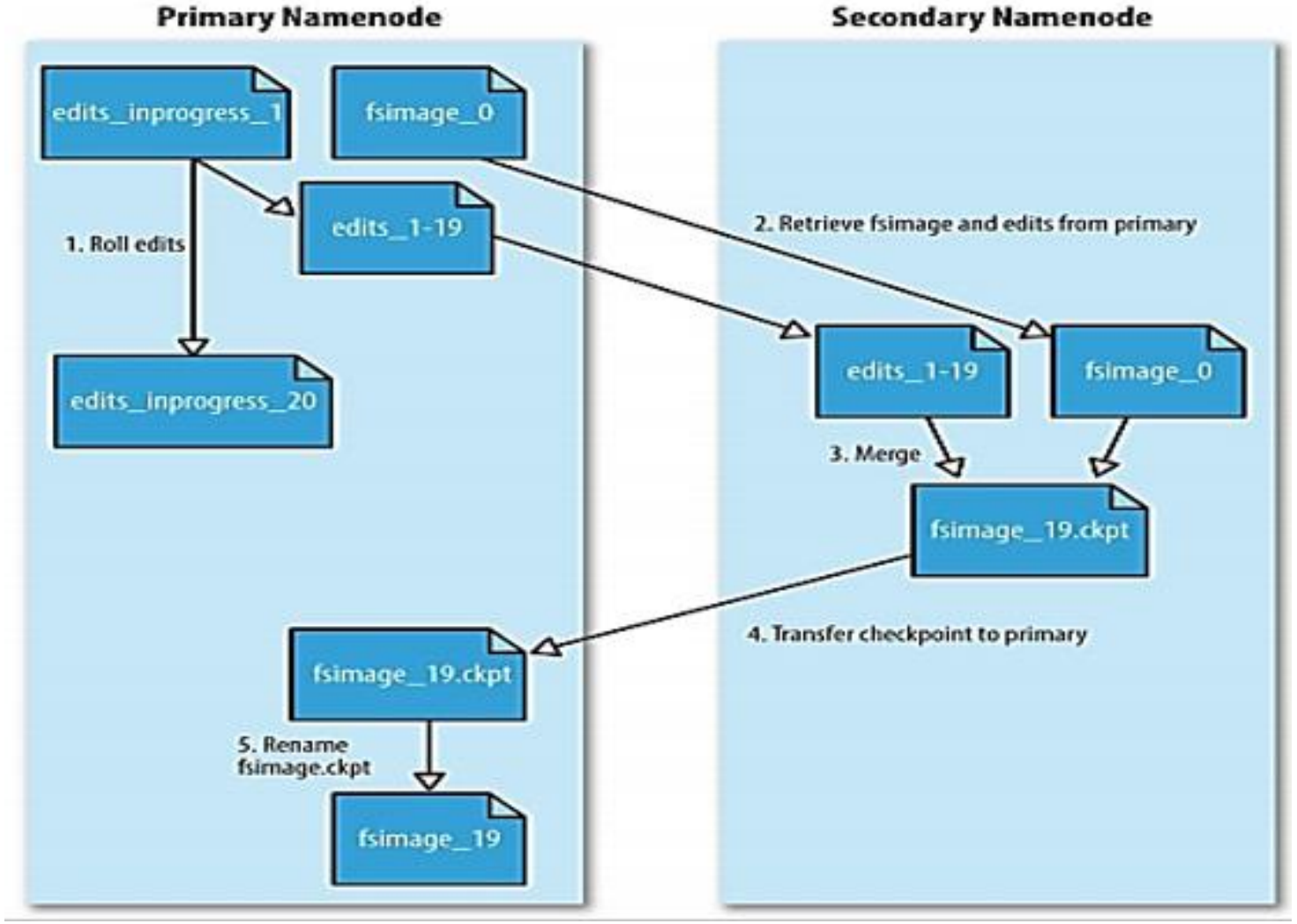
NameNode (Contd.)

2. Edits

- When any operation takes place in HDFS, the directory structure gets modified
- These modifications are stored in the memory as well as in the edits files (edits files are stored on the hard disk)
- If the existing fsimage file gets merged with edits, we'll get an updated fsimage file
- This process is called "checkpointing" and is carried out by the Secondary NameNode. It takes the fsimage and edits files from the NameNode and returns updated fsimage file after merging.

Checkpointing

ACADGILD



Questions and Answers

ACADGILD

1. Why do we need to merge fsimage and edits?

   Ans: fsimage contains the HDFS metadata and edits contain the changes in the HDFS metadata. To get the exact view of the HDFS structure, both must be merged. At the time when a NameNode restarts, it merges both of them, as it needs to store this information in memory. Since edits can become very large with time, a NameNode's startup time can become very lengthy. So, it is good to have them merged periodically.

2. Do the fsimage or edits files store mapping for blocks and datanodes containing them?

   Ans: No, this mapping information is not stored in the fsimage or edits files. These files store the directory structure and other metadata about the filesystem. Mapping information is stored in the memory of a NameNode (i.e. RAM), so that requests can be served quickly.

Questions and Answers (Contd. 1)

ACADGILD

3. How does NameNode know which DataNodes store which blocks of a file?

Ans: When a cluster or NameNode starts, DataNodes join the cluster. In this process they send the information about the blocks stored by them to NameNode. NameNod keeps this information in memory. Moreover, DataNodes keep on sending the same information periodically to NameNode, so that namenode remains in sync with lates mapping.

4. Why updates in HDFS file system structure of file system metadata are not directly merged with fsimage file?

Ans: fsimage has a larger size, so merging it with updates is time-consuming as well as space consuming (as all contents of fsimage is loaded in memory first and then updates are applied). So, updates are placed separately.

Questions and Answers (Contd. 2)

ACADGILD

5. Why is a different machine (Secondary NameNode) needed for merging the edits with the fsimage (i.e., Checkpointing)?

   Ans: Merging of edits with fsimage is time consuming as well as resource consuming. Since NameNode should be highly available, this checkpoiniting task is taken care of by another daemon called Secondary NameNode. NameNode does contain the merged information in memory, but if it has to write its content to the file, the process will be time consuming. This will make the cluster irresponsive for some time.

6. How often does checkpointing takes place?

   Ans: Checkpointing takes place usually every hour or once edits become large (i.e., 64 MB or more). But these parameters are configurable.

7. Why NameNode should not fail easily?

   Ans: In Hadoop 1.x, NameNode is the single point of failure. If it fails, we won't be able to access the cluster.

Questions and Answers (Contd. 3)

ACADGILD

8. Can the secondary NameNode act as NameNode?

Ans: The secondary NameNode is not designed to act as a NameNode. It keeps the recent fsimage of the NameNode. But it can work as a NameNode after the fsimage is merged with the latest edits and the blocks' mapping is received after communication with the DataNodes. In case NameNode fails suddenly and we are unable to access its edits, then the latest updates over the HDFS file structure will be lost as the secondary NameNode stores only the recent fsimage but not the edits.

Safe Mode:

- During the start up, the NameNode loads the file system state from the fsimage and the edits log file.
- It then waits for DataNodes to report their blocks. During this time, the NameNode stays in the Safemode.
- Safemode for the NameNode is essentially a read-only mode for the HDFS cluster, where it does not allow any modifications to the file system or blocks.

Replica Placement

ACADGILD

- How does the NameNode choose which DataNodes to store the replicas on?
- Placing all replicas on a single node incurs the lowest write bandwidth penalty (since the replication pipeline runs on a single node).
- But this offers no real redundancy (if the node fails, the data for that block is lost).
- Also, the read bandwidth is high for the off-rack reads.
- At the other extreme, placing replicas in different data centres may maximize redundancy, but at the cost of the write bandwidth.
- Hadoop's default strategy is to place the first replica on the same node as the client.
- For clients running outside the cluster, a node is chosen at random.
- The system tries not to pick nodes that are too full or too busy.

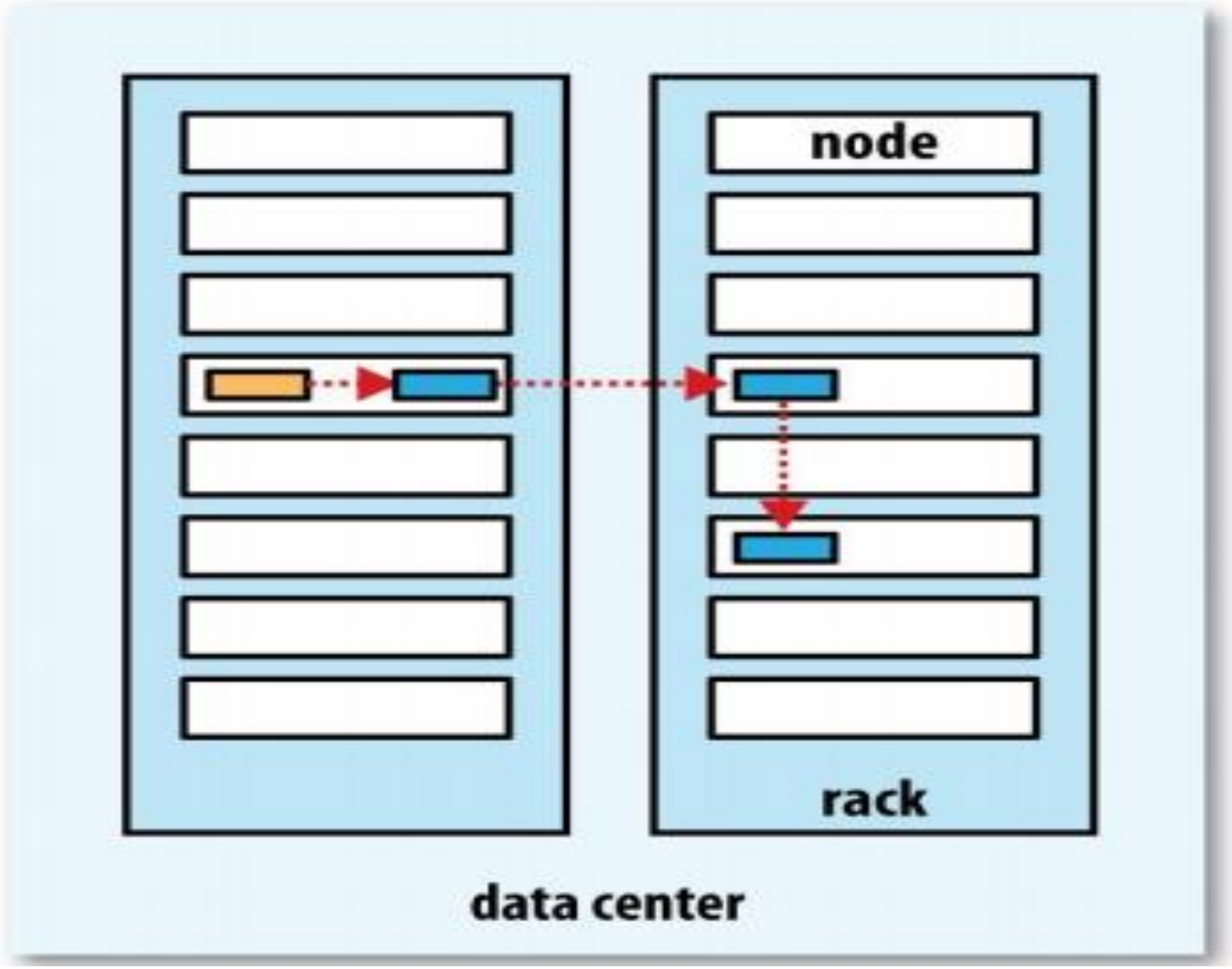
Replica Placement (Contd. 1)

ACADGILD

- The second replica is placed on a different rack from the first (off-rack), and is chosen at random.
- The third replica is placed on the same rack as the second, but on a different node chosen at random.
- Further replicas are placed on random nodes in the cluster, although the system tries to avoid placing too many replicas on the same rack.

Replica Placement (Contd. 2)

ACADGILD



Benefits of Replica Placement and Rack Awareness

ACADGILD

This strategy gives a good balance among:

- Reliability (blocks are stored on two racks, so data is available even in case of node or rack failure)
- Write bandwidth (writes only have to traverse a single network switch)
- Read performance (there's a choice between two racks to read from)
- Block distribution across the cluster (clients only write a single block on the local rack)

Balancer:

- A tool that analyzes block placement and re-balances data across the DataNodes.

Questions and Answers

ACADGILD

- For replication factor 3, the default strategy of block storage is:

1. One block on client, two on different machines of a different rack
2. Two blocks on client, one on a different machine of the same rack
3. One block on client, one on machine of a different rack, and the last on a machine of a different data centre.
4. All three blocks are stored randomly on different machines.

Ans: One block on client, two on different machines of different racks.

Questions and Answers (Contd.)

ACADGILD

- Which of the following is not a responsibility of DataNode?

1. Store actual data blocks of file in HDFS on its own local disk
2. Sends signals to NameNode periodically
3. Calculate the checksum of data blocks
4. Contain HDFS metadata

Ans: Contains the HDFS metadata

URI, URL, and URN

ACADGILD

URI (Uniform Resource Identifier)

- Uniquely identifies a resource along with its scheme

URL (Uniform Resource Locator)

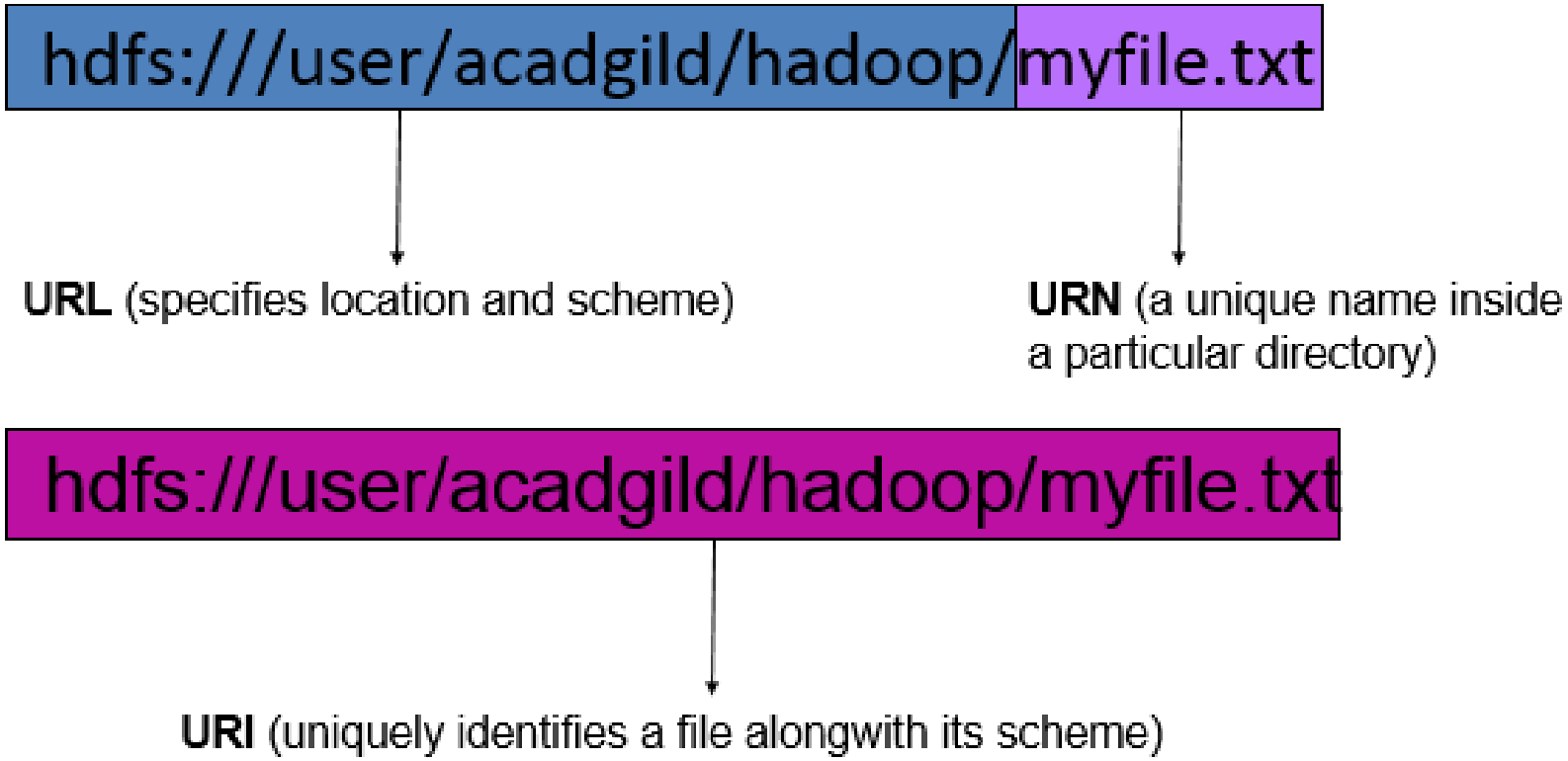
- Defines the network location and a scheme about how a resource will be obtained

URN (Uniform Resource Name)

- Identifies a resource by the name in a given namespace

URI, URL, and URN (Contd.)

ACADGILD



HDFS Commands

ACADGILD

Read Commands

- cat
- checksum
- ls
- text

HDFS Commands (Contd.)

Write Commands

- appendToFile
- copyFromLocal
- put
- moveFromLocal
- copyToLocal
- get
- cp
- mkdir
- mv
- rm

Questions and Answers

ACADGILD

- Which of the following is not an HDFS write command?

1. put
2. moveToLocal
3. touchz
4. Checksum

Ans: checksum

Problems with HDFS in Hadoop 1.x

ACADGILD

- NameNode is a single point of failure. Failure of NameNode, leads to an inaccessible Hadoop cluster.
- NameNode must be a high-end machine which is less prone to failure.
- A secondary NameNode cannot act as a NameNode.
- In the HDFS Federation, we use multiple NameNodes, each storing the fsimage, edits, and block mappings of different portions of the HDFS.
- In HDFS High Availability, multiple namenodes are used in the Active-Standby mode with the shared edits to handle NameNode failure.

HDFS Federation (Included in Hadoop 2.x)

ACADGILD

- Let HDFS contain two sub-directories in the root directory.

  /usr

  /share

- In the HDFS Federation, there are multiple NameNodes, each storing the metadata and block mapping of files and directories contained in particular sub-directories.

- The list of sub-directories managed by a NameNode is called a namespace volume.

- Blocks for files belonging to a namespace is called a block pool.

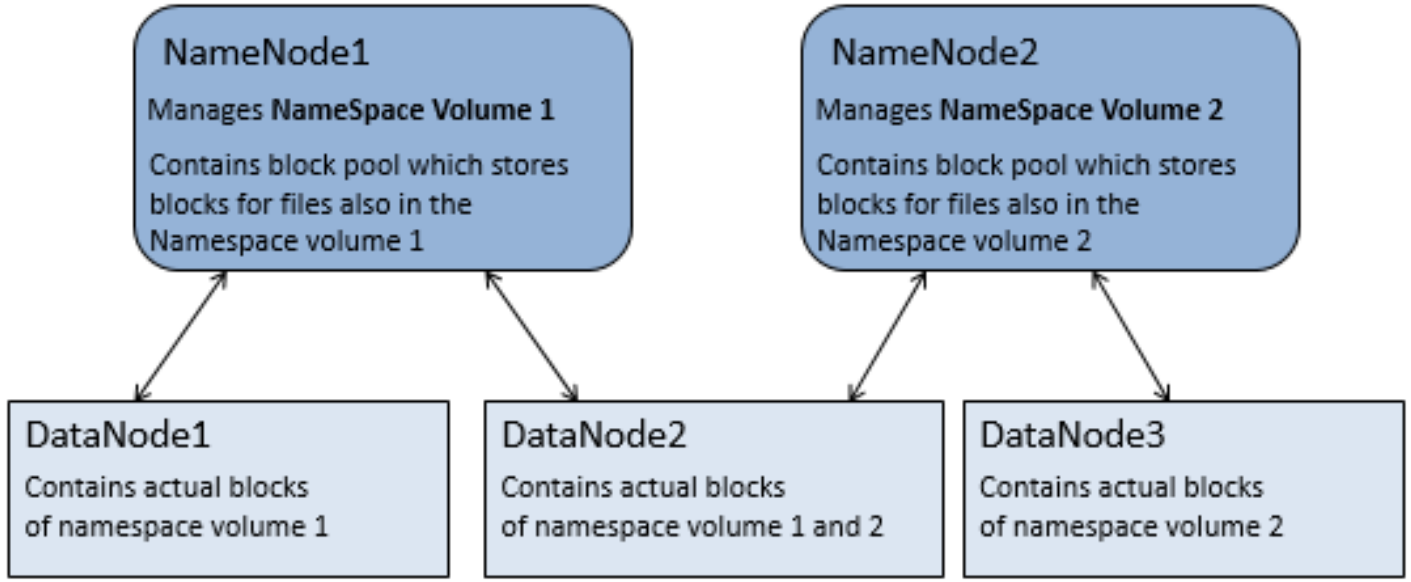
  For example, here we can have two namenodes, one for storing the metadata and block mapping for namespace volume /usr and one for /share.

- This way, if one NameNode fails, a namespace volume managed by an other namenode is still accessible. So the entire cluster doesn't become inaccessible.

HDFS Federation

ACADGILD

- Let,
  - Sub-directories and files inside /usr constitute NameSpace Volume 1
  - Sub-directories and files inside /share constitute NameSpace Volume 2
- One DataNode can contain blocks of different namespace volumes
- So, namespace volumes are divided among NameNodes, but not among DataNodes.



High Availability (included in Hadoop 2.x)

ACADGILD

- There is a pair of NameNodes in an **active-standby** configuration
- In the event of failure of an active NameNode, the standby takes over its duties without a significant interruption.

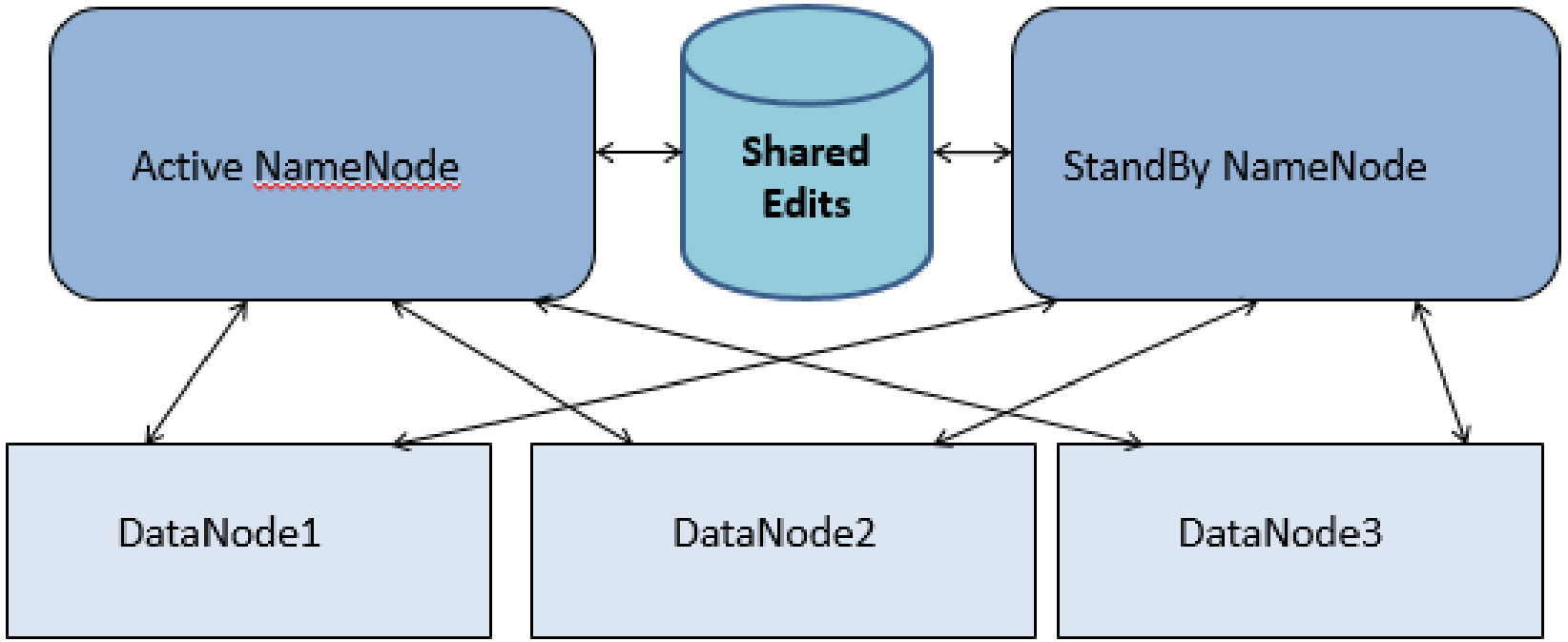
Architectural Changes:

- The NameNodes must use highly-available shared storage to share the edit log. Edit logs are read by the StandbyNameNode when it takes the responsibility of the ActiveNameNode.
- DataNodes must send block reports to both the NameNodes because of the block mappings.
- The secondary NameNode's role is subsumed by the StandBy NameNode. StandbyNameNode takes periodic checkpoints of the active NameNode.
- Checkpointing is done by the StandBy NameNode.

High Availability

ACADGILD

- Each DataNode reports block stored by it to both the NameNodes



Questions and Answers

ACADGILD

- We need to increase the availability of the HDFS cluster by using multiple NameNodes. But our NameNodes do not have sufficient RAM to store the block mapping of the entire HDFS. What approach will fit our need?

1. HDFS Federation
2. HDFS High Availability
3. Either HDFS Federation or HDFS High Availability
4. None of the above

Ans: HDFS Federation

Questions and Answers (Contd.)

ACADGILD

- Identify the incorrect statement(s)

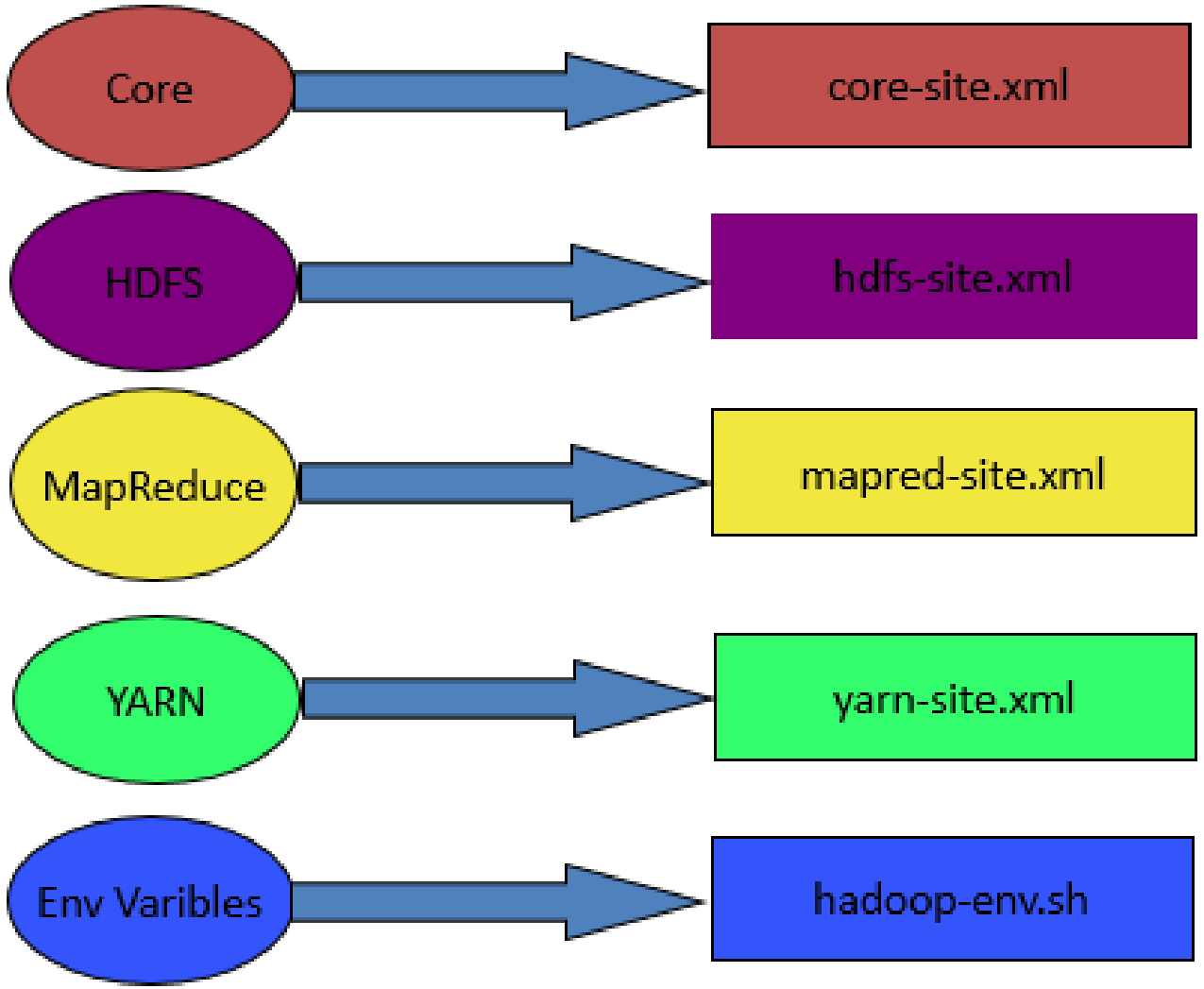
1. In HDFS High Availability, checkpointing is done by the secondary Namenode
2. In HDFS Federation and HDFS High Availability, a DataNode may report to multiple NameNodes
3. In HDFS Federation, one DataNode must report to a single NameNode
4. In HDFS High Availability, cluster is available even if all the NameNodes go down

Ans: 3. In HDFS Federation, one DataNode must report to a single NameNode

4. In HDFS High Availability, cluster is available even if all the NameNodes go down

Configuration Files in Hadoop

ACADGILD



Configuration Files in Hadoop (contd.)

ACADGILD

- Hadoop comes up with default configurations in *-default.xml.

  For example: hdfs-default.xml, yarn-default.xml, etc.

- Default configurations are overridden in *-site.xml.

  For example: hdfs-site.xml, yarn-site.xml, etc.

HDFS Configurations

ACADGILD

- dfs.replication
- dfs.replication.max
- dfs.namenode.replication.min
- dfs.blocksize
- dfs.heartbeat.interval
- dfs.hosts
- dfs.hosts.exclude
- dfs.bytes-per-checksum

HDFS Configurations (Contd.)

ACADGILD

- dfs.namenode.checkpoint.dir
- dfs.namenode.checkpoint.edits.dir
- dfs.namenode.checkpoint.period
- dfs.namenode.checkpoint.txns
- dfs.namenode.checkpoint.check.period
- dfs.namenode.checkpoint.max-retries
- dfs.namenode.num.checkpoints.retained

Core Configurations

ACADGILD

Core-default.xml

- Property: io.bytes.per.checksum

Default: 512

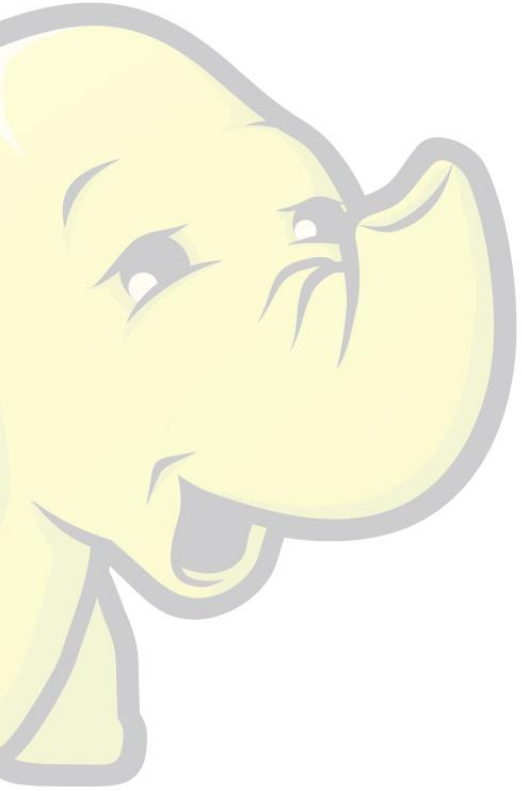
Description: The number of bytes per checksum.

- Property: fs.defaultFS

Default: file:///

Description: The name of the default file system.

A URI whose scheme and authority determine the FileSystem implementation.



THANK YOU

Email us at: support@acadgild.com