BIG DATA ASSIGNMENT

Name – Vedang Kulkarni Roll no. – DS5A-1942

***What is Hadoop Zookeeper?***

Zookeeper is an open-source Apache project that offers support for naming, synchronization, and gathering administrations over large clusters in a distributed system through circulated frameworks. The objective is to make these frameworks simpler to deal with. It provides a centralized platform for maintaining configurations across group services.

It can be thought of as a tool to extract information from a distributed platform of large clusters of systems.

***Why is it needed?***

Coordinating and working with distributed platforms with huge volumes of data and information are prone to cause errors in file handling and processing of data. As the number of clusters increases, the chances of causing errors also increase.

With Zookeeper, the implementation of distributed applications over large platforms becomes easier to work with.

Zookeeper uses various synchronizations to deal with such problems of distributed applications.

***Zookeeper Architecture -***

Zookeeper has a file system that has nodes instead of directories to store data.

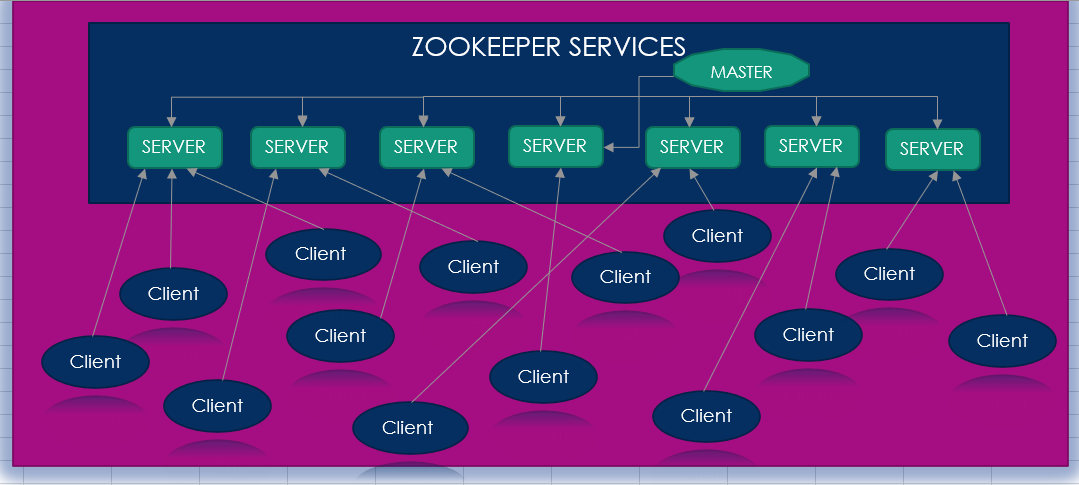
Zookeeper works on a client-server architecture where clients are users of a service and servers provide that service to clients. Several servers are ensembled to provide services to the clients. One client is connected to at least one server node. These nodes in Zookeeper are known as Znodes. Among the servers, a master server is randomly selected by the system to address the proper functioning of services.

All read and write operations are directed through the master server to ensure systematic stepwise operations. All servers are interconnected to ensure data literacy.

There are two types of znodes -

Persistence node – These nodes are independent of the session or process performed in the zookeeper service. These are also called permanent nodes. Persistence nodes are internal to the zookeeper service.

Ephemeral node – These nodes are dependent on the session and process performed on the server. These are temporary nodes that get removed upon client disconnection or process completion.



*In this way, Zookeeper manages and coordinates large clusters of machines.*

***How does it work?***

Step-1: As soon as the cluster of servers starts, the client nodes get connected to the servers.

Step-2: After that, each client node gets connected to at least one server node, that server node can be either a master node or any other server node.

Step-3: On a successful connection, the server assigns a unique ID to the client node for that session. The client node is then identified by that unique ID throughout the session.

Step-4: If there is an unsuccessful connection, server and client nodes try and reevaluate the connection by sending acknowledgments.

Step-5: The client nodes maintain connections with the servers through acknowledgments during the real-time execution of processes.

Step-6: Finally, as per the need client performs read and write operations and later, stores the data as per requirements.

***Features of Zookeeper –***

There are four most implicit features of Zookeeper –

1. The status of each node is updated after every operation to maintain stepwise and updated information on the operations performed.
2. Real-time updating allows easier management of clusters.
3. Unique identification of client node helps in proper functioning and reduces allocation errors.
4. Automated failure recovery helps in reducing data loss and ensures proper functionality across server nodes.

***Benefits –***

Five benefits make our working with Zookeeper easier. These are –

1. Simplicity – Hierarchical serialization of processes makes it simple to organize and manage processes across multiple servers.
2. Reliability – Proper synchronization of server processes using unique identification allows greater reliability.
3. Order – Ordered messages and sequential consistency helps keep every process on track.
4. Speed – The higher speed of process execution ensures more credibility.
5. Scalability – Performance of the system can be enhanced by deploying more clusters to improve scalability and efficiency.

***Uses Cases –***

Zookeeper is used for the following use case scenarios –

1. Configuration management
2. Naming services
3. Queuing messages
4. Notification system management
5. Synchronization