

PRACTICAL-4

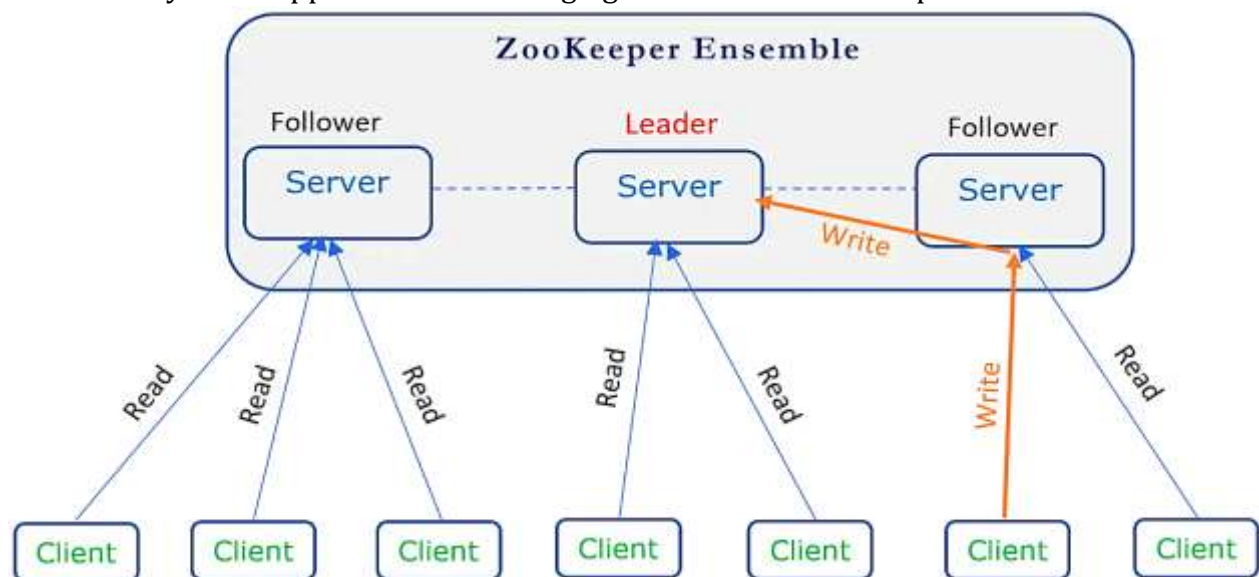
Perform the Zookeeper commands in CloudxLab and Windows

ZOOKEEPER

- Zookeeper in Hadoop can be considered a centralized repository where distributed applications can put data into and retrieve data from.
- It makes a distributed system work together as a whole using its synchronization, serialization, and coordination goals.
- Zookeeper can be thought of as a file system where we have nodes that store data instead of files or directories that store data.

Why do we need Zookeeper in the Hadoop?

Distributed applications are difficult to coordinate and work with, as they are much more error prone due to huge number of machines attached to network. As many machines are involved, race condition and deadlocks are common problems when implementing distributed applications. Race condition occurs when a machine tries to perform two or more operations at a time and this can be taken care by serialization property of ZooKeeper. Deadlocks are when two or more machines try to access same-shared resource at the same time. More precisely, they try to access each other's resources, which leads to lock of system as none of the system is releasing the resource but waiting for other system to release it. Synchronization in Zookeeper helps to solve the deadlock. Another major issue with distributed application can be partial failure of process, which can lead to inconsistency of data. Zookeeper handles this through atomicity, which means either whole of the process will finish or nothing will persist after failure. Thus Zookeeper is an important part of Hadoop that take care of these small but important issues so that developer can focus more on functionality of the application. Following figure shows the Zookeeper architecture.



Writes in Zookeeper

All the writes in Zookeeper go through the Master node, thus it is guaranteed that all writes will be sequential. On performing write operation to the Zookeeper, each server attached to that client persists the data along with master. Thus, this makes all the servers updated about the data. However, this also means that concurrent writes cannot be made. Linear writes guarantee can be problematic if Zookeeper is used for write dominant workload. Zookeeper in Hadoop is ideally used for coordinating message exchanges between clients, which involves less writes and more reads. Zookeeper is helpful till the time the data is shared but if application has concurrent data writing then Zookeeper can come in way of the application and impose strict ordering of operations.

Reads in Zookeeper

Zookeeper is best at reads as reads can be concurrent. Concurrent reads are done as each client is attached to different server and all clients can read from the servers simultaneously, although having concurrent reads leads to eventual consistency as master is not involved. There can be cases where client may have an outdated view, which gets updated with a little delay.

What is ZNode?

ZNode can be thought of as directories in the file system. The only difference is that Znodes have the capability of storing data as well. ZNode's task is to maintain the statistics of the structure and version numbers for data changes. Data 'read' and 'write' happen in its entirety, as Zookeeper does not allow partial 'read' and 'write'.

Types of Zookeeper Nodes

PERSISTENCE ZNODE

- Once created these Znodes will be there forever in the Zookeeper.
- To remove these Znodes, you need to delete them manually(use delete operation).
- we can store any config information or any data that needs to be persistent.
- All servers can consume data from this Znode.
- *Note: If no flag is passed, by default persistent znode is created.*

SEQUENTIAL ZNODE

- Sequential znodes can be either persistent or ephemeral.
- When a new znode is created as a sequential znode, then ZooKeeper sets the path of the znode by attaching a 10 digit sequence number to the original name.
- For example, if a znode with path /myapp is created as a sequential znode, ZooKeeper will change the path to /myapp0000000001 and set the next sequence number as 0000000002. If two sequential znodes are created concurrently, then ZooKeeper never uses the same number for each znode.
- Sequential znodes play an important role in Locking and Synchronization.

EPHEMERAL ZNODE

- These znodes are automatically deleted by the Zookeeper, once the client that created it, ends the session with zookeeper.
- Ephemeral znodes play an important role in Leader election.

ZOOKEEPER Command Line Interface – CLOUDXLAB

ZooKeeper Command Line Interface (CLI) is used to interact with the ZooKeeper ensemble, which lets you perform simple, file-like operations. It is useful for debugging purposes. To perform ZooKeeper CLI operations, first start your ZooKeeper server and Client.

STEP-1: Open webconsole (in cloudxlab)

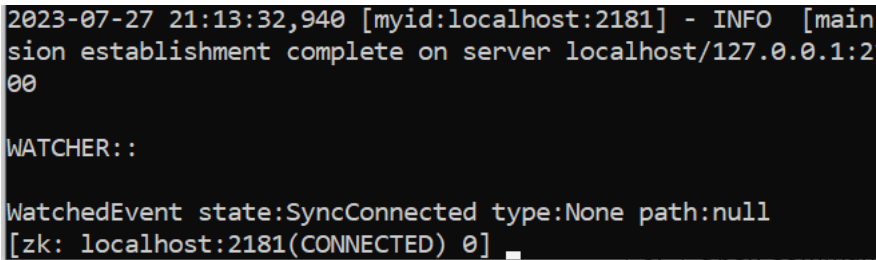
STEP-2: Type the command “zookeeper-client” and enter.

Perform following Commands

ls /	See the list of ZNodes at the top level
ls /brokers	See the children of znodes under a parent node (Ex. Brokers is parent node name)
create /path “data”	Create Znodes (Ex. create /OneZnode “HelloZookeeper-app”)
create -s /path “data”	Create Sequential znode (Ex. create -s /TwoZnode “HelloZookeeper-app”) NOTE: Sequential znodes guaranty that the znode path will be unique. ZooKeeper ensemble will add sequence number along with 10-digit padding to the znode path. For example, the znode path /myapp will be converted to /myapp0000000001 and the next sequence number will be /myapp0000000002. If no flags are specified, then the znode is considered as persistent.
create -e /path “data”	Create an Ephemeral Znode (Ex. create -e /ThreeZnode “uvpce”) NOTE: Ephemeral znodes (flag: e) will be automatically deleted when a session expires or when the client disconnects. Remember when a client connection is lost; the ephemeral znode will be deleted. You can try it by quitting the ZooKeeper Client and then re-opening the Client.
get /path	Get Data (Ex. get /OneZnode) get /TwoZnode0000000023 (To access a sequential znode, you must enter the full path of the znode.) NOTE: It returns the associated data of the znode and metadata of the specified znode. You will get information such as when the data was last modified, where it was modified, and information about the data. This CLI is also used to assign watches to show notification about the data. NOTE: Below show the output details of “get” command and description. my_data: This line of text is the data that we stored in the znode. cZxid=0x8: The zxid (ZooKeeper Transaction Id) of the change that caused this znode to be created. ctime=Mon Nov 30 18:41:06 IST 2015: The time when this znode

	<p>was created.</p> <p>mZxid=0x8: The zxid of the change that last modified this znode.</p> <p>mtime=Mon Nov 30 18:41:06 IST 2015: The time when this znode was last modified.</p> <p>pZxid=0x8: The zxid of the change that last modified children of this znode.</p> <p>cversion=0: The number of changes to the children of this znode.</p> <p>dataVersion=0: The number of changes to the data of this znode.</p> <p>aclVersion=0: The number of changes to the ACL of this znode.</p> <p>ephemeralOwner=0x0: The session id of the owner of this znode if the znode is an ephemeral node. If it is not an ephemeral node, it will be zero.</p> <p>dataLength=7: The length of the data field of this znode.</p> <p>numChildren=0: The number of children of this znode.</p>
get /path [watch] 1	<p>Create Watch on znode</p> <p>get /OneZnode 1</p> <p>NOTE: Watches show a notification when the specified znode or znode's children data changes. You can set a watch only in get command. The output is similar to normal get command, but it will wait for znode changes in the background.</p> <p>The objective of watches is to get notified when znode changes in some way. Watchers are triggered only once. If you want recurring notifications, you will have re-register the watcher.</p>
set /path "data"	<p>set /FirstZnode "Data-updated"</p> <p>NOTE: Set the data of the specified znode. Once you finish this set operation, you can check the data using the get CLI command. If watch is active on node and you are trying to change the data then set command will show the watch notification.</p>
create /parent/path "data"	<p>Create Children / Sub-znode</p> <p>create /OneZnode/Child1 "child one"</p> <p>NOTE: Creating children is similar to creating new znodes. The only difference is that the path of the child znode will have the parent path as well.</p>
stat /path	<p>Check Status</p> <p>stat /OneZnode</p> <p>NOTE: Status describes the metadata of a specified znode. It contains details such as Timestamp, Version number, ACL, Data length, and Children znode.</p>
rmr /path	<p>Remove a Znode</p> <p>rmr /OneZnode</p> <p>NOTE: Delete (delete /path) command is similar to remove command, except the fact that it works only on znodes with no children.</p>
quit	Quit

ZOOKEEPER SETUP ON WINDOWS 10

1.	Download following for Zookeeper setup
	JDK: https://www.oracle.com/in/java/technologies/downloads/#jdk20-windows
	Zookeeper: https://www.apache.org/dyn/closer.lua/zookeeper/zookeeper-3.7.1/apache-zookeeper-3.7.1-bin.tar.gz
2.	Install JDK20 (It is compulsory)
	Default JDK installation path: C:\Program Files\Java
	Set System Environment variables JAVA_HOME=C:\Program Files\Java\jdk-20 Path=C:\Program Files\Java\jdk-20\bin
3.	Extract the “apache-zookeeper-3.7.1-bin.tar.gz” and copy the extracted folder (In my case folder name is “zookeeper-3.7.1”) in following location: C:\Tool\zookeeper-3.7.1
4.	Go to “C:\Tool\zookeeper-3.7.1” and create folder named “data” inside this path.
5.	Go to “C:\Tool\zookeeper-3.7.1\conf” and rename file “zoo_sample.cfg” to “zoo.cfg” Open “zoo.cfg” in Notepad Set the parameter “dataDir=C:/Tool/zookeeper-3.7.1/data”
6.	Set System Environment variables ZOOKEEPER_HOME= C:\Tool\zookeeper-3.7.1
7.	Open Command Prompt as Administrator and type zkserver (Server in running mode) C:>zkserver
8.	Open another Command Prompt as Administrator and run following command to start the zookeeper CLI C:>cd C:\Tool\zookeeper-3.7.1\bin C:\Tool\zookeeper-3.7.1\bin>zkCli OUTPUT: 
9.	Run the following Zookeeper command inside the zookeeper CLI
	[zk: localhost:2181(CONNECTED) 2] ls / [zookeeper] [zk: localhost:2181(CONNECTED) 4] create /First "Hello" Created /First [zk: localhost:2181(CONNECTED) 5] ls / [First, zookeeper] [zk: localhost:2181(CONNECTED) 6] get /First Hello [zk: localhost:2181(CONNECTED) 7] create -s /SecondSEQ "Sequential znode" Created /SecondSEQ0000000001 [zk: localhost:2181(CONNECTED) 8] create -e /ThirdEphemeral "Ephemeral znode"

	Created /ThirdEphemeral [zk: localhost:2181(CONNECTED) 10] get /SecondSEQ0000000001 Sequential znode [zk: localhost:2181(CONNECTED) 11] get /ThirdEphemeral Ephemeral znode [zk: localhost:2181(CONNECTED) 19] create /First/Child1 "child" Created /First/Child1 [zk: localhost:2181(CONNECTED) 23] stat /First [zk: localhost:2181(CONNECTED) 20] get -w /First //Watch event run when data changed Hello [zk: localhost:2181(CONNECTED) 24] set /First "Hi" WATCHER:: [zk: localhost:2181(CONNECTED) 25] WatchedEvent state:SyncConnected type:NodeDataChanged path:/First [zk: localhost:2181(CONNECTED) 25] quit
10.	Stop zkserver "Press Control+C" Write "Y" press Enter