Hands-On Exercise: Working with HDFS

In this hands-on exercise you will explore several ways to access and manage HDFS. You will also create a host template and use it to add a new worker host to your cluster.

IMPORTANT: This exercise depends on Hands-On Exercise: Configuring a Hadoop Cluster. Be sure to complete that exercise before continuing.

All steps in this exercise that use a terminal window should be run on `cmhost`.

Create a host template for worker hosts

In the next section, you will add a new worker host to the cluster. As you have probably noticed, we have reached critical data size on cmhost. We will move the DataNode off of cmhost. Before you do that, you will create a `worker` host template to use for the new host that includes the HDFS DataNode role.

1. In Cloudera Manager, select **Hosts** > **Host Templates** and click **Create**.

   a. Enter the template name **Worker**

   b. Select the following role groups to include in role groups in the template:

   - **HDFS DataNode** in **DataNode Default Group**
   - **YARN NodeManager** in the **NodeManager Default Group**

   c. Click **Create**.

   Verify that there are two different "... Default Group" roles in the `Worker` template.

Add a worker host to the cluster

The `worker-3` host was provisioned earlier, but was not added to the cluster. In this step, you will add it to the cluster using the `Worker` host template you created above.

2. Go to **Hosts** > **All Hosts**.

---

3. Click the **Add Hosts** button.

---

4. Select **Add hosts to cluster** option, and then click **Continue** to start the Add Hosts wizard.

---

5. In the **Specify Hosts** step, **Search** for the `worker-3` host. Confirm it is selected, then click **Continue.**

---

6. In the **Select Repository** screen, choose the **Custom Repository** option and enter **http://cmhost:8060/cm7.3.1/**. Then click click **Continue**.

---

7. Make sure that the **Manually manage JDK** option is selected, then click **Continue**.

---

8. Configure the new host's login credentials.

   a. Make sure that **Login To All Hosts As** is changed to **Another user** and enter **training**.

   b. Choose authentication method **All hosts accept the same private key**, then:

      i. Browse to `/home/training/.ssh` if needed.

      Tip: You may need to right-click and choose **Show hidden files** to see the `.ssh` directory.

      ii. Select the `id_rsa` file and click **Open**.

      iii. Click **Continue.**

---

9. The wizard will proceed to install the Cloudera Manager Agent. This may take a few minutes to complete.

   When it completes, click **Continue**.

---

10. The **Select Host Template** screen appears showing the available templates.

---

11. Choose the **Worker** template you created above and click **Continue**.

---

12. The role instances identified by the template you created will be applied to the new host.

    This may take a few minutes to complete.

---

13. After all parcels are activated on the new host, click **Continue** if needed. Typically it will move to the next page automatically.

---

14. On **Inspect hosts** screen, confirm that all validations succeeded. Click **Continue**.

---

15. After all the steps that start roles and deploy client configurations complete, click **Continue**.

16. Click **Finish**.

17. From the Status page of the Cloudera Manager Home page, click on any one of the **Stale Services** icons, and **Restart Stale Services** (be sure to redeploy client configurations), then click **Restart Now**. The cluster restart may take up to five minutes to complete.

18. After cicking **Finish**, go to the HDFS **Status** tab.

    Note that the configured capacity has increased to about 340 GB because you just added a fourth DataNode to your cluster.

Decommission and delete the `cmhost` DataNode and NodeManager roles

In the exercise environment, `cmhost` is currently serving as a YARN NodeManager and HDFS DataNode. Having a utility node serve these roles is not a good practice. Adding `worker-3` as a new DataNode and NodeManager added more capacity to the system, so you can now remove those roles from `cmhost`.

19. Go to the HDFS **Instances** tab.

20. Check the box next to **DataNode** that is deployed to `cmhost`.

21. Select **Actions for Selected > Decommission**.

22. In the confirmation dialog, click **Decommission** and wait for the action to complete, then click **Close**.

23. Verify on the HDFS **Instances** tab that the status of the DataNode role on `cmhost` is now **Decommissioned**.

24. Check the box next to **DataNode** that is deployed to `cmhost` again.

25. Select **Actions for Selected > Delete**, then confirm that you want to delete the role on the host.

26. Go to the **YARN** service page, then follow the steps above to decommission and delete the NodeManager role from `cmhost` as well.

---

27. Go to the Cloudera Manager home page and restart the stale services. Be sure to re-deploy stale configurations.

Confirm HDFS Processes and Roles

28. Confirm that all HDFS processes are running.

a. From the Cloudera Manager, click **HDFS** to go to the HDFS service page, then go to the **Instances** tab.

b. Notice that that the HDFS daemons (`NameNode`, `SecondaryNameNode`, and multiple `DataNodes`) are running on various hosts in the cluster. There is also a `Gateway` role deployed to the `cmhost` machine.

---

29. Use the search box in the HDFS **Configuration** tab, and search for **block size**.

Note that the HDFS Block Size setting defaults to 128MB.

View the HDFS File Browser in Cloudera Manager

30. On the HDFS service page, select the **File Browser** tab.

---

31. Click into the `user` directory.

Notice the user directories exist for the OS users that run the CDP services that exist in the cluster, such as `hive` and `hue`. However, there is no user space defined yet in HDFS for the `training` user. In the next section you will create the `training` user directory.

View HDFS User Folders

32. In the cmhost terminal, enter

```
$ hdfs dfs -ls /user/
```

Notice that there is no `/user/training` directory in HDFS.

---

33. Using the `hdfs` user, create a home directory for the training user in HDFS. Then set the correct permissions.

In the `cmhost` terminal, enter:

```
$ sudo -u hdfs hdfs dfs -mkdir /user/training
$ sudo -u hdfs hdfs dfs -chown training /user/training
```

---

34. Check to ensiure the folder has now been created:

```
$ hdfs dfs -ls /user/
```

Manage the HDFS Superuser Group

35. Try creating a new directory in HDFS:

```
$ hdfs dfs -mkdir /testdir
```

Note: The command fails with a "Permission denied" error, because the `training` user does not have permission to write to the root directory of HDFS.

---

36. In Cloudera Manager, go to the HDFS configuration tab.

---

37. Search for **dfs.permissions**. Note that permission-checking is enabled.

---

38. Note that `dfs.permissions.superusergroup` defaults to a group named `supergroup`. Users in `supergroup` are HDFS "superusers"—that is, they have full permissions to perform any operation in HDFS.

The `supergroup` OS user group does not exist yet on the NameNode host. Create the group, and add the `training` OS user to the group on the NameNode host (`master-1`):

```
$ ssh training@master-1 sudo groupadd supergroup
$ ssh master-1 sudo usermod -aG supergroup training
```

Note: If these commands fail with `command not found`, use these commands instead:

```
$ ssh training@master-1 sudo /usr/sbin/groupadd supergroup
$ ssh master-1 sudo /usr/sbin/usermod -aG supergroup training
```

> For simplicity, these exercises add the `supergroup` group only to the NameNode host. In a production environment, the cluster would likely be configured to get user and group information from an external source, such as LDAP or Active Directory. You would add the group in one place and it would propagate throughout the cluster in a consistent way.

---

39. Confirm the group was created.

```
$ ssh master-1 groups training
```

The command should show:

```
training : training wheel supergroup
```

---

40. Run the `hdfs dfsadmin` command to refresh the HDFS user group mapping cache.

```
$ hdfs dfsadmin -refreshUserToGroupsMappings
```

Add directories and files using the HDFS CLI

41. Re-run the command that failed due to a permission error earlier.

```
$ hdfs dfs -mkdir /testdir
```

---

42. Verify the directory was created.

```
$ hdfs dfs -ls /
```

---

43. Create three test files on the local `cmhost` file system and copy them into HDFS.

```
$ cd /tmp
$ touch testfile1 testfile2 testfile3
$ hdfs dfs -put testfile* /testdir/
```

---

44. Verify the files were placed in HDFS.

```
$ hdfs dfs -ls /testdir
```

---

45. Delete one of the files from HDFS.

```
$ hdfs dfs -rm /testdir/testfile1
```

Note the info message saying that the file was moved to the trash.

---

46. Delete another one of the files, this time skipping the trash.

```
$ hdfs dfs -rm -skipTrash /testdir/testfile2
```

---

47. Try to delete the directory.

```
$ hdfs dfs -rmdir /testdir
```

The command did not work, because there is still a file in the directory.

---

48. Delete the directory recursively, which deletes the directory and all its contents.

```
$ hdfs dfs -rm -r /testdir
```

---

49. Empty the `training` user's trash.

```
$ hdfs dfs -expunge
```

---

50. The `hdfs` user owns the NameNode process and has superuser permissions to HDFS, even without being in the group specified by the `dfs.permissions.supergroup` setting.

Run the `mkdir` command as the `hdfs` user.

```
$ sudo -u hdfs hdfs dfs -mkdir /weblogs
```

Tip: Remember that, in the exercise environment, the `training` OS user has full passwordless `sudo` permissions, so you can run any command as any user. This would not be typical for an end user in a production cluster.

---

51. Verify ownership of the new directory:

```
$ hdfs dfs -ls /
```

---

52. Change the owner of the `weblogs` directory to be the `training` user, then verify the new ownership.

```
$ sudo -u hdfs hdfs dfs -chown training /weblogs
$ hdfs dfs -ls /
```

Add web server log data to HDFS

In this section you will upload to HDFS a set of files containing log output from a web server.

53. Find the size of the data on the Linux filesystem.

```
$ du -sm ~/training_materials/admin/data/weblogs
```

The command shows the size in MB.

Find the number of files in the `weblogs` directory.

```
$ ls ~/training_materials/admin/data/weblogs | wc -l
```

---

54. Add the files to the new directory in HDFS.

```
$ hdfs dfs -put ~/training_materials/admin/data/weblogs/* \
  /weblogs/
```

The command will take a moment to copy the files from the `cmhost` machine's local filesystem to the cluster HDFS storage (the machines where the DataNode roles are running).

---

55. View a summary of how the weblogs are being stored.

```
$ hdfs fsck /weblogs -blocks
```

Tip: The "Total size" statistic shown in the results of the command is displayed in bytes.

---

56. Use the `fsck` utility to discover more detail about the files, including which DataNodes host copies (replicas) of the blocks that make up each file in the weblogs dataset.

```
$ hdfs fsck /weblogs -blocks -files -racks
```

Load the Ngrams dataset into HDFS

The Google Ngrams Dataset

In this section, you will load data from the Google Books Ngrams dataset, which is licensed under a Creative Commons Attribution

3.0 Unported License https://creativecommons.org/licenses/by/3.0/. The American English 1grams dataset is the one you will use.

It contains all the words extracted from American English publications from the Google Books corpus.

See https://aws.amazon.com/datasets/google-books-ngrams/ and http://storage.googleapis.com/books/ngrams/books/datasetsv2.html for more details.

57. In Cloudera Manager, go to the HDFS **Status** tab.

---

58. In the **HDFS Summary** panel, note the configured capacity.

It should show a total of 218.8 GB of storage capacity.

---

59. Use the Linux du command to view the size of the ngrams dataset. For this exercise, you will be using a subset of the dataset: the files ending with a through e.

```
$ du -h /ngrams/unzipped/*[a-e]
```

The approximate size of the sizes of the files ending with a through e is 7GB.

---

60. Copy the dataset into HDFS. This may take several minutes.

```
$ hdfs dfs -mkdir /tmp/ngrams
$ hdfs dfs -put /ngrams/unzipped/*[a-e] /tmp/ngrams/
```

---

61. Open the NameNode web UI at http://master-1:9870 and click on the **Datanodes** page. Note the number of blocks stored on each worker host.

Refresh the page a few times to see that the number of blocks on each DataNode is increasing.

---

62. Return to the Cloudera Manager HDFS **Status** tab.

Note the **Total Bytes Written Across DataNodes** and **HDFS Capacity** charts. They should reflect the process of uploading the ngram data.

---

63. When the put action has completed, refresh the HDFS **Status** tab.

Configured Capacity should now show an additional 14 GB used of the total 170GB of space available. The new value reflects the prior level plus the 7GB you just uploaded, stored with 2x replication = 14GB space used to store the ngrams subset data.

Explore replication and block information

64. Return to the NameNode web UI and choose **Utilities** > **Browse the file system**. Open the `/tmp/ngrams` directory and note the information available for each block of data, including the file size, replication factor, and block size. Notice that the file sizes shown in the **Size** column are larger than the block sizes.

---

65. Click on the first file in the list: `googlebooks-eng-all-1gram-20120701-a`. This opens the **File information** pop-up window.

Notice that the size of the first block of the file (**Block 0**) is 134,217,728 bytes (128 MB). This is because the HDFS block size on your cluster is 128 MB.

---

66. Click on the **Block information** drop-down menu and switch the view to last block in the file (**Block 13**).

Notice that the last block is smaller than the first: about 57 MB.

In the **Availability** section, take note of which DataNodes hold a copy of each block. In a later step, you will locate the file on disk, so you need to know which DataNodes have a copy.

---

67. Select the value of the `Block ID` field for **Block 13** and copy it to the clipboard. You will need this value for the next step in this exercise.

---

68. Locate the HDFS block on the Linux file systems on one of the NameNodes in the cluster that has a copy of the file.

a. In the Cloudera Manager HDFS **Configuration** tab, search for "data directory." Note that the DataNode Data Directory is `/dfs/dn`.

b. In the terminal window, use the Linux `find` command to locate a copy of the block on disk.

Note: In the following command, replace `datanode` with the hostname of one of the two DataNodes that has a copy of the block. Also replace **block id** with

the actual Block ID you copied from the NameNode Web UI. Be sure to keep the `'` and `*` characters as shown.

```
$ ssh training@datanode sudo find \
  /dfs/dn -name '*block-id*' -ls
```

c. Verify that two files with the block ID you copied appear in the `find` command output—one file with extension `.meta`, and another file without any extension.

The `.meta` file is a metadata file that contains checksums for sections of the block.

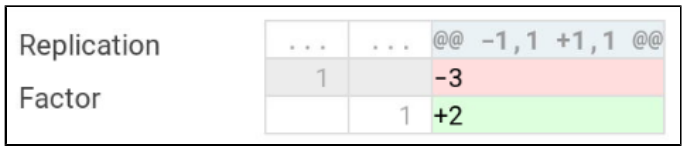
Increase the HDFS replication factor for new files

69. In the HDFS **Configuration** tab, click on **History and Rollback** on the right side of the page.

---

70. Locate the **Modified Replication Factor** revision listed towards to bottom

There may be more than one revision with that label. Use the **Details** link to find the one in which you previously changed the replication factor from the default value (3) to 2, as shown below.



---

71. After confirming in the **Revision Details** that you have selected the correct revision, click **Revert Configuration Changes**.

---

72. Redeploy the configurations and restart the stale services.

---

73. In a `cmhost` terminal, confirm the replication factor of new files.

```
$ echo "test file contents" > testfile
$ hdfs dfs -put testfile /tmp/
$ hdfs dfs -ls /tmp/testfile
```

The result of the last command should show that the replication factor of the new file is 3:

```
-rw-r--r--   3 training supergroup
```

Increase the replication for selected existing files

Changing the HDFS replication factor configuration does not automatically change the replication of existing files. In this section, you will manually change the replication of the files in the `weblogs` HDFS directory.

74. Verify that the current web log files are replicated twice.

```
$ hdfs dfs -ls /weblogs
```

---

75. Increase the replication factor of the files in the `weblogs` directory to 3.

```
$ hdfs dfs -setrep -R 3 /weblogs
```

---

76. Rerun the command to list the files to confirm that the replication factor has been correctly set to 3.

This is the end of the exercise.