Hands-On Exercise: Data Ingest with Hadoop Tools

Files and Data Used in This Exercise

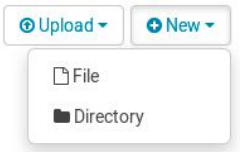
Exercise directory	$ADIR/exercises/data_ingest
Files	$ADIR/data/access.log
MySQL tables	employees, customers, products,
	orders, order_details
Hive/Impala tables	customers, products, orders,
	order_details

In this exercise, you will practice using the Hadoop command line utility to interact with Hadoop's Distributed Filesystem (HDFS) and use Sqoop to import tables from a relational database to HDFS.

Exploring HDFS using the Hue File Browser

1. Launch the hands-on environment.
2. In Firefox, click the **Hue** bookmark in the bookmark toolbar (or type **http://localhost:8889/hue/home** into the address bar and press the Enter key.)

3. After a few seconds, you should see Hue's home screen. Enter in the username: **cloudera** and password: **cloudera** fields, and then click the **Sign In** button. **Note:** You can close the tour that appears. These exercises will present the features you need for this course.
4. Click the menu icon (☰) to the left of the Hue icon, then click **Browsers > Files**. The File Browser displays your HDFS home directory. This directory does not yet contain any files or directories.
5. Create a temporary sub-directory: select the **+New** menu and click **Directory**.



6. Enter directory name **test** and click the **Create** button. Your home directory now contains a directory called test.
7. Click **test** to view the contents of that directory; currently it contains no files or subdirectories.
8. Upload a file to the directory. Start by selecting **Upload**.
9. Click **Select files** to bring up a file browser. By default, the /home/training/Desktop folder displays. Click the home directory button (**training**) then navigate to the course data directory: training_materials/data.
10. Choose any of the data files in that directory, *except* access.log. (Hue limits file uploads to 64MB, so you must upload larger files using the command line.) Then click the **Open** button.

In Case of Error

If Hue displays the error message Error: IOException: Failed to find datanode, this indicates that one or more of the required services has failed and needs to be restarted.

11. The file you selected will be loaded into the current HDFS directory. Click the filename to see the file's contents. Because HDFS is designed to store very large files, Hue will not display the entire file, just the first page of data. You can use the scrollbar to see more of the data.

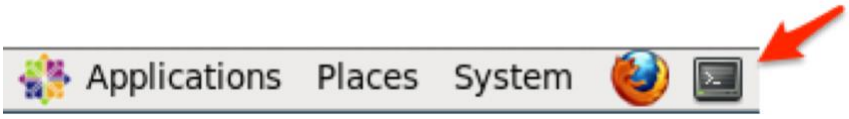
12. Return to the test directory by clicking **Back** on the left side of the main panel.

13. Above the list of files in your current directory is the full path of the directory you are currently displaying. You can click on any directory in the path, or on the first slash (/) to go to the top level (root) directory. Click **training** to return to your home directory.

14. Delete the temporary test directory you created, including the file in it, by selecting the checkbox next to the directory name then clicking the **Move to trash** button. Confirm that you want to delete by clicking **Yes**. Your home directory now shows a .Trash subdirectory.

Exploring HDFS Using the Command Line

15. You can use the hdfs dfs command to interact with HDFS from the command line. Close or minimize Firefox, and then open a terminal window by clicking the icon in the system toolbar:



16. In the terminal window, enter:

$ hdfs dfs

This displays a help message describing all subcommands associated with hdfs dfs.

17. Run the following command:

$ hdfs dfs -ls

The -ls flag says to list the contents of a directory. Since no path is specified, hdfs dfs assumes you are referring to your home directory. This output will show list the .Trash subdirectory, since that's the current contents of your home directory.

18. Now run the following command:

$ hdfs dfs -ls /

This lists the contents of the HDFS root directory, specified by the path /. One of the directories listed is /user. Each user on the cluster has a "home" directory below / user corresponding to his or her user ID.

19. Note the /analyst directory. Most of your work in this course will be in that directory. Try creating a temporary subdirectory in /analyst:

$ hdfs dfs -mkdir analyst/test1

20. Next, add a web server log file to this new directory in HDFS:

$ hdfs dfs -put $ADIR/data/access.log analyst/test1/

Overwriting Files in Hadoop

Unlike the UNIX shell, Hadoop won't overwrite files and directories. This feature helps protect users from accidentally replacing data that may have taken hours to produce. If you need to replace a file or directory in HDFS, you must first remove the existing one. Please keep this in mind in case you make a mistake and need to repeat a step during the Hands-On Exercises.

21. To remove a file:

```
$ hdfs dfs -rm analyst/example.txt
```

22. To remove a directory and all its files and subdirectories (recursively):

```
$ hdfs dfs -rm -r analyst/example/
```

23. Verify the last step by listing the contents of the analyst/test1 directory. You should observe that the access.log file is present and occupies 106,339,468 bytes of space in HDFS:

```
$ hdfs dfs -ls analyst/test1
```

24. Remove the temporary directory and its contents:

```
$ hdfs dfs -rm -r analyst/test1
```

Importing Database Tables into HDFS with Sqoop

Dualcore stores information about its employees, customers, products, and orders in a MySQL database. In the next few steps, you will examine this database before using Sqoop to import its tables into HDFS.

25. Install databse in to MySQL through:

```
$ cd <exercise files path>/data/mysql/
$ mysql --user=root --password=cloudera -e "source dbsetup.sql"
```

26. In a terminal window, log in to MySQL and select the analyst_dualcore database:

```
$ mysql --user=root --password=cloudera analyst_dualcore
```

Note: The exercise environment uses MariaDB as its MySQL database. If you are used to MySQL, the only difference you should notice is the prompt, which says MariaDB instead of mysql.

26. Next, list the available tables in the analyst_dualcore database (MariaDB> represents the MySQL client prompt and is not part of the command):

```
MariaDB> SHOW TABLES;
```

27. Review the structure of the employees table and examine a few of its records:

```
MariaDB> DESCRIBE employees;
MariaDB> SELECT emp_id, fname, lname, state, salary FROM employees LIMIT 10;
```

28. Exit MySQL by typing **quit**, and press the Enter key:

```
MariaDB> quit
```

29. Create a database with the name “analyst”, in a terminal window, write:

```
$ hive
hive> create database analyst;
```

30. Run the following command, which imports the employees table into the /analyst/dualcore directory in HDFS, using tab characters to separate each field:

```
$ sqoop import \
--connect jdbc:mysql://localhost/analyst_dualcore \
--username root --password cloudera \
--fields-terminated-by '\t' \
--warehouse-dir analyst/dualcore \
--table employees
--hive-import
--create-hive-table
--hive-table analyst.employees
```

31. Revise the previous command and import the customers table into HDFS.

32. Revise the previous command and import the products table into HDFS.

33. Revise the previous command and import the orders table into HDFS.

34. Next, import the order_details table into HDFS. The command is slightly different because this table only holds references to records in the orders and products table, and it lacks a primary key of its own. Consequently, you will need to specify the --split-by option and instruct Sqoop to divide the import work among tasks based on values in the order_id field. An alternative is to use the -m 1 option to force Sqoop to import all the data with a single task, but this would significantly reduce performance.

```
$ sqoop import \
--connect jdbc:mysql://localhost/analyst_dualcore \
--username root --password cloudera \
--fields-terminated-by '\t' \
--warehouse-dir /analyst/dualcore \
--table order_details \
--split-by order_id
--hive-import
--create-hive-table
--hive-table analyst.order_details
```

35. Finally, verify the above using the command:

```
$ hdfs dfs -ls /analyst/dualcore
```

Check that you have five subdirectories, named after the five MySQL tables that you imported. If any are missing, go back to the appropriate step and try again. (Don't just use the up arrow and re-run the command—if you made a mistake in typing, you need to find and fix the error.) If you like, you can also list the contents of those directories to see that data has been transferred.