Lab: Getting Started with Pig

About this Lab

Objective:	Use Pig to navigate through HDFS and explore a dataset.
File locations:	/root/hdp/pigandhive/labs/Lab5.1
Successful outcome:	You will have a couple of Pig programs that load the White House visitors' data, with and without a schema, and store the output of a relation into a folder in HDFS.
Before you begin:	Your HDP 2.6 cluster should be up and running within your VM.
Related lesson:	***Introduction to Pig***

Lab Steps

1) View the Raw Data

If not already done, open a Terminal in your VM and type "ssh sandbox". Change directories to the `/root/hdp/pigandhive/labs/Lab5.1` folder:

```
cd ~/hdp/pigandhive/labs/Lab5.1
```

Unzip the archive in the `/root/hdp/pigandhive/labs/Lab5.1` folder, which contains a file named `whitehouse_visits.txt` that is quite large:

```
unzip whitehouse_visits.zip
```

View the contents of this file:

```
tail whitehouse_visits.txt
```

This publicly available data contains records of visitors to the White House in Washington, D.C.

2) Load the Data into HDFS

a. Start the Grunt shell:

```
# pig
```

From the Grunt shell, make a new directory in HDFS named `whitehouse`:

```
grunt> mkdir whitehouse
```

Use the `copyFromLocal` command in the Grunt shell to copy the `whitehouse_visits.txt` file to the whitehouse folder in HDFS, renaming the file visits.txt. (Be sure to enter this command on a single line):

```
grunt> copyFromLocal
/root/hdp/pigandhive/labs/Lab5.1/whitehouse_visits.txt
whitehouse/visits.txt
```

d. Use the `ls` command to verify that the file was uploaded successfully:

```
grunt> ls whitehouse
hdfs://sandbox.hortonworks.com:8020/user/root/whitehouse/visits.tx
t<r 1>183292235
```

3) Define a Relation

You will use the TextLoader to load the `visits.txt` file.

Note

TextLoader simply creates a tuple for each line of text, and it uses a single chararray field that contains the entire line. It allows you to load lines of text and not worry about the format or schema yet.
Define the following `LOAD` relation:

```
grunt>A = LOAD '/user/root/whitehouse/' USING TextLoader();
```

b. Use `DESCRIBE` to notice that `A` does not have a schema:

```
grunt> DESCRIBE A;
Schema for A unknown.
```

We want to get a sense of what this data looks like. Use the `LIMIT` operator to define a new relation named `A_limit` that is limited to 10 records of `A`.

```
grunt> A_limit = LIMIT A 10;
```

d. Use the `DUMP` operator to view the `A_limit` relation.

Each row in the output will look similar to the following and should be 10 arbitrary rows from visits.txt:

```
grunt> DUMP A_limit;

(WHITLEY,KRISTY,J,U45880,,VA,,,,,10/7/2010 5:51,10/9/2010
10:30,10/9/2010 23:59,,294,B3,WIN,10/7/2010
5:51,B3,OFFICE,VISITORS,WH,RES,OFFICE,VISITORS,GROUP TOUR
,1/28/2011,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
```

,,,)

4) Define a Schema

Load the White House data again, but this time use the PigStorage loader and also define a partial schema:

```
grunt> B = LOAD '/user/root/whitehouse/visits.txt'
USING PigStorage(',') AS (
   lname:chararray,
   fname:chararray,
   mname:chararray,
   id:chararray,
   status:chararray,
   state:chararray,
   arrival:chararray
);
```

Commented [AK2]:

b. Use the DESCRIBE command to view the schema:

```
grunt> describe B;
   {lname: chararray,fname: chararray,mname: chararray,id:
chararray,status: chararray,state: chararray,arrival:
chararray}
```

5) The `STORE` Command

Enter the following `STORE` command, which stores the `B` relation into a folder named `whouse_tab` and separates the fields of each record with tabs:

```
grunt> store B into 'whouse_tab' using PigStorage('\t');
```

Verify that the `whouse_tab` folder was created: `grunt> ls whouse_tab`

You should see two map output files.

View one of the output files to verify they contain the `B` relation in a tab-delimited format:

```
grunt> fs -tail whouse_tab/ part-v000-o000-r-00000
```

d. Each record should contain seven fields. What happened to the rest of the fields from the raw data that was loaded from `whitehouse/visits.txt`?

Answer: They were simply ignored when each record was read in from HDFS.

6) Use a Different Storer

In the previous step, you stored a relation using PigStorage with a tab delimiter. Enter the following command, which stores the same relation but in a `JSON` format:

```
grunt> store B into 'whouse_json' using JsonStorage();
```

Verify that the `whouse_json` folder was created:

```
grunt> ls whouse_json
```

View one of the output files:

```
grunt> fs -tail whouse_json/part-v000-o000-r-00000
```

Notice that the schema you defined for the B relation was used to create the format of each JSON entry:

Result

You have now seen how to execute some basic Pig commands, load data into a relation, and store a relation into a folder in HDFS using different formats.