1. Apache Pig: LOAD and STORE

Load

In Apache Pig, the `LOAD` statement is used to load data from various sources into Pig relations. It is one of the most fundamental operations in Pig Latin, which is the scripting language for Pig. The general syntax for the `LOAD` statement in Pig is as follows:

```
<alias> = LOAD '<input>' [USING function] [AS schema];
```

Here, the parts of the syntax have the following meanings:

- `<alias>`: The name you want to give to the relation being created. It acts as a reference to the data being loaded.
- `LOAD`: This keyword is used to indicate that data is being loaded into a relation.
- `<input>`: This represents the source of the data, such as a file path or an HDFS (Hadoop Distributed File System) location.
- `USING function`: (Optional) This is used when you want to specify a specific function to process the data during the load operation, such as a custom loader or a built-in loader like PigStorage.
- `AS schema`: (Optional) This is used to specify the schema of the data being loaded. If not provided, Pig will attempt to infer the schema.

Here is an example of the `LOAD` statement in Pig:

```
-- Loading data from a CSV file with a custom delimiter
my_data = LOAD 'hdfs://mydata/data.csv' USING PigStorage(',') AS (id:int,
name:chararray, age:int);
```

In this example, `my_data` is the name of the relation being created. The data is loaded from the file `data.csv` located at the HDFS path `hdfs://mydata/`. The data is assumed to be comma-separated, and the schema is explicitly specified as having three fields: `id` as an integer, `name` as a character array, and `age` as an integer.

Example 1

1. First let's make a file called `pigfile.txt`. This is what it looks like:

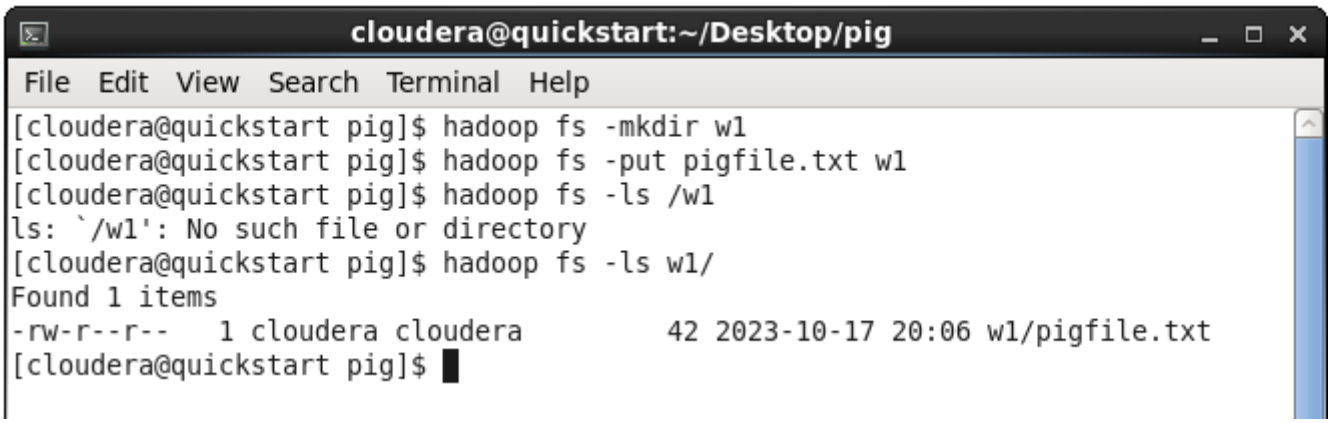
```
1,2,3,4,
5,6,7,8,
```

```
9,10,11,12,
13,14,15,16
```

2. Now let's put the file in HDFS.

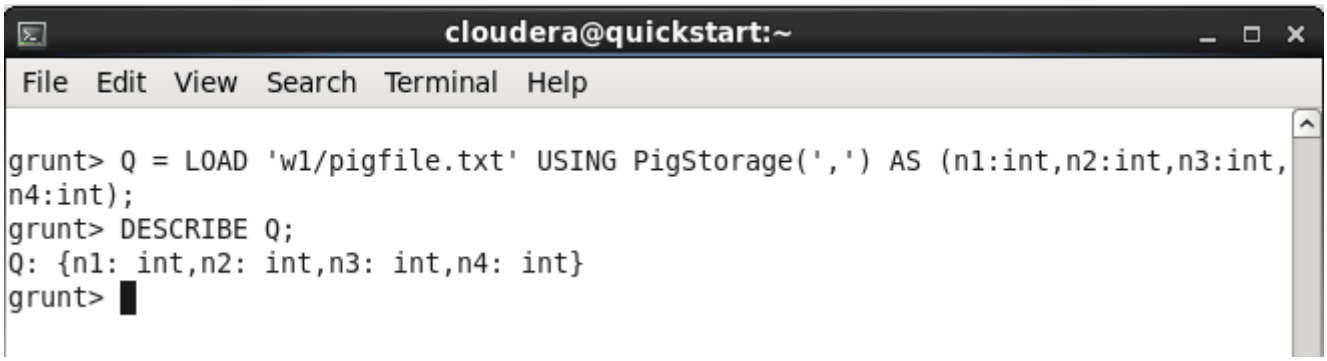
   `hadoop fs -mkdir w1`

   `hadoop fs -put pigfile.txt w1`



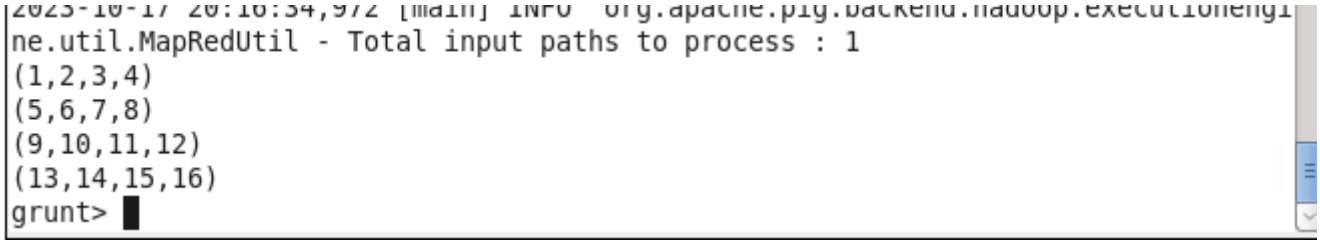
4. Let's launch pig in MapReduce mode. Using `pig`

5. `Q = LOAD 'w1/pigfile.txt' USING PigStorage(',') AS (n1:int,n2:int,n3:int,n4:int);`



6. DESCRIBE Q;

7. DUMP Q;



STORE

In Apache Pig, the `STORE` command is used to store the results of a Pig Latin script into an output location, such as a file, HDFS, or HBase. The general syntax for the `STORE` command in Pig is as follows:

```
STORE <relation> INTO '<output>' [USING function];
```

Here, the components of the syntax have the following meanings:

- `<relation>`: The name of the relation whose data you want to store.
- `INTO`: This keyword is used to indicate that the data is being stored.
- `<output>`: This represents the location where the data will be stored, such as a file path or an HDFS (Hadoop Distributed File System) location.
- `USING function`: (Optional) This is used when you want to specify a specific function to process the data during the store operation, such as a custom storage function or a built-in storage function like PigStorage.

Here is an example of the `STORE` command in Pig:

```
-- Storing the data in a relation to a CSV file
STORE my_data INTO 'hdfs://output/results.csv' USING PigStorage(',');
```

In this example, `my_data` is the name of the relation whose data will be stored. The data will be stored in a file called `results.csv` located at the HDFS path `hdfs://output/`. The data will be stored in a comma-separated format because the `PigStorage` function with the delimiter ',' is used.

Example 1 Continued

`STORE Q INTO 'pigoutput' USING PigStorage('|');`
==Note: In the above case `pigoutput` directory wasn't created before hand. But created by the command itself.

```
Output(s):
Successfully stored 4 records (39 bytes) in: "hdfs://quickstart.cloudera:8020/us
er/cloudera/pigoutput"

Counters:
Total records written : 4
Total bytes written : 39
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1697597774621_0002


2023-10-17 20:23:17,746 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - Success!
grunt>
```

Let's check the file exists on HDFS:

```
[cloudera@quickstart pig]$ hdfs dfs -ls /user/cloudera
Found 2 items
drwxr-xr-x   - cloudera cloudera          0 2023-10-17 20:23 /user/cloudera/pigo
utput
drwxr-xr-x   - cloudera cloudera          0 2023-10-17 20:06 /user/cloudera/w1
[cloudera@quickstart pig]$ hdfs dfs -ls /user/cloudera/pigoutput
Found 2 items
-rw-r--r--   1 cloudera cloudera          0 2023-10-17 20:23 /user/cloudera/pigo
utput/_SUCCESS
-rw-r--r--   1 cloudera cloudera         39 2023-10-17 20:23 /user/cloudera/pigo
utput/part-m-00000
[cloudera@quickstart pig]$
```

```
[cloudera@quickstart pig]$ hdfs dfs -cat /user/cloudera/pigoutput/part-m-00000

1|2|3|4
5|6|7|8
9|10|11|12
13|14|15|16
[cloudera@quickstart pig]$
[cloudera@quickstart pig]$
```