Fetching JSON Data

Using Pig & Hive

Version 2.0

Fetching JSON Data

Table of Contents

Fetching JSON Data using Pig & Hive

Problem Statement:

Here, we are fetching the JSON data with the help of sample dataset. It is explained as:

- Fetching JSON data using Pig
- Fetching JSON data using Hive

Important Links:

Pig Installation guide:
https://edureka.wistia.com/medias/lpb6yiupps

Hive Installation guide:
http://www.edureka.co/blog/apache-hive-installation-on-ubuntu

Edureka VM Installation:
Please refer to Installation guide section present in the LMS for accessing the Edureka VM Installation Guide.

Hive-serdes-1.0-SNAPSHOT JAR:
https://edureka.wistia.com/medias/gsamkn57is/download?media_file_id=67382949

Tools and Technologies used:

- Pig
- Hive

Dataset for Pig:

Let us consider a sample dataset as in the below box.

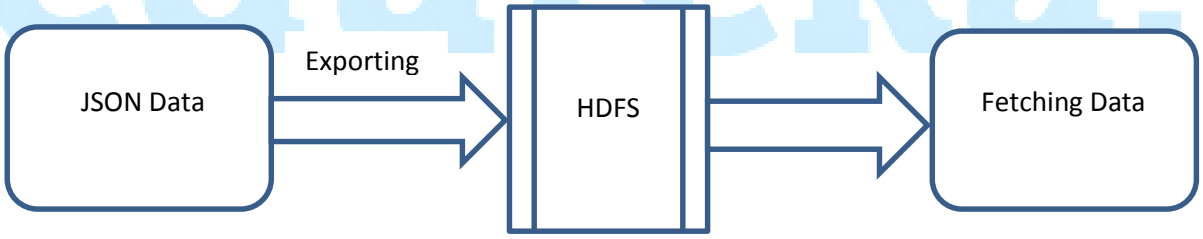
```
{"recipe":"Tacos","ingredients":[{"name":"Beef"},{"name":"Lettuce"},{"name":
"Cheese"}],"inventor":{"name":"Alex","age":25}}
{"recipe":"TomatoSoup","ingredients":[{"name":"Tomatoes"},{"name":"Milk"}],
"inventor":{"name":"Steve","age":23}}
```

Dataset Description:

All the tags of the above sample is given below:

- Recipe
- Ingredients
  - Name
- Inventor
  - Name
  - Age

Dataflow Diagram:



Implementation:

After moving data to HDFS, we just loaded the data and dumping it.

Let us see how to do that:

First let us log into pig shell.

Command: pig

Command: second_table = LOAD '/home/edureka/Desktop/first_table.json' USING JsonLoader('recipe:chararray, ingredients: {(name:chararray)}, inventor: (name:chararray, age:int)');

```
grunt> second_table = LOAD '/home/edureka/Desktop/first_table.json' USING JsonLoader('recipe:chararray, ingredients:
{(name:chararray)}, inventor: (name:chararray, age:int)');
2015-01-06 09:28:34,278 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapreduce.job.counters.limit
nters.max
grunt>
```

Now we are dumping the data.

Command: dump second_table;

```
2015-01-06 09:30:34,873 [main] INFO  org.apache
(Tacos,{(Beef),(Lettuce),(Cheese)},(Alex,25))
(TomatoSoup,{(Tomatoes),(Milk)},(Steve,23))
grunt>
```

Dataset for Hive:

Let us consider a sample dataset as in the below box.

{"id":"AP112","info":{"OS":"IOS 7","model_name":"Apple Iphone5","emi_no":"APXX156734","registered_date":"12/11/2013"}}

{"id":"AP133","info":{"OS":"IOS 8","model_name":"Apple Iphone4","emi_no":"APXX156756","registered_date":"02/09/2014"}}

{"id":"G156","info":{"OS":"Android 3","model_name":"Samsung S5","emi_no":"GAXXX84754","registered_date":"10/10/2012"}}

{"id":"AP1904","info":{"OS":"IOS 8","model_name":"Apple Iphone5","emi_no":"APXX64314","registered_date":"07/11/2010"}}

{"id":"AP2345","info":{"OS":"IOS 8","model_name":"Apple Iphone6","emi_no":"APXX64029","registered_date":"03/11/2013"}}

{"id":"AP8906","info":{"OS":"IOS 7","model_name":"Apple Iphone4","emi_no":"APXX64123","registered_date":"03/11/2014"}}

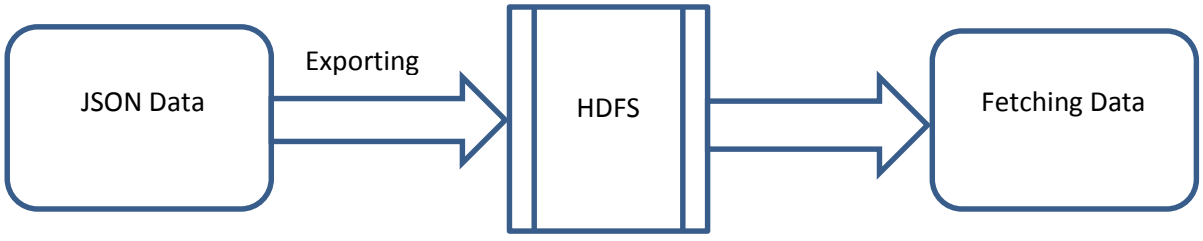
{"id":"G671","info":{"OS":"Android 5","model_name":"Samsung S4","emi_no":"GAXXX98765","registered_date":"12/11/2013"}}

Dataset Description:

All the tags of the above sample is given below:

- id
- info
  - OS
  - model_name
  - emi_no
  - registered_date

Dataflow Diagram:



Implementation:

After moving data to HDFS, we are creating the table and loading the data into the table.

Let us see how to do that:

First let us log into hive shell, use the created database and created the table.

Command: hive

Command: create database jsonFile;

Command: use jsonFile;

Command: **ADD jar /home/edureka/Downloads/hive-serdes-1.0-SNAPSHOT.jar;**

Command: create table devices (id string, info struct< OS:string, model_name:string, emi_no:string, registered_date:string >) row format serde 'com.cloudera.hive.serde.JSONSerDe';

Command: load data local inpath '/home/edureka/Desktop/device_info.json' overwrite into table devices;

```
hive> create database jsonFile;
OK
Time taken: 0.045 seconds
hive> use jsonFile;
OK
Time taken: 0.014 seconds
hive> ADD jar /home/edureka/Downloads/hive-serdes-1.0-SNAPSHOT.jar;
Added /home/edureka/Downloads/hive-serdes-1.0-SNAPSHOT.jar to class path
Added resource: /home/edureka/Downloads/hive-serdes-1.0-SNAPSHOT.jar
hive> create table devices (id string,  info struct< OS:string, model_name:string, emi_no:string, registered_date:string >)
row format serde 'com.cloudera.hive.serde.JSONSerDe';
OK
Time taken: 0.066 seconds
hive> load data local inpath '/home/edureka/Desktop/device_info.json' overwrite into table devices;
Copying data from file:/home/edureka/Desktop/device_info.json
Copying file: file:/home/edureka/Desktop/device_info.json
Loading data to table jsonfile.devices
rmr: DEPRECATED: Please use 'rm -r' instead.
Deleted hdfs://localhost:8020/user/hive/warehouse/jsonfile.db/devices
Table jsonfile.devices stats: [numFiles=1, numRows=0, totalSize=1776, rawDataSize=0]
OK
Time taken: 0.284 seconds
```

Now we are displaying all the fields of the table.

Command: select * from devices;

```
hive> select * from devices;
OK
AP112   {"os":"IOS 7","model_name":"Apple Iphone5","emi_no":"APXX156734","registered_date":"12/11/2013"}
AP133   {"os":"IOS 8","model_name":"Apple Iphone4","emi_no":"APXX156756","registered_date":"02/09/2014"}
G156    {"os":"Android 3","model_name":"Samsung S5","emi_no":"GAXXX84754","registered_date":"10/10/2012"}
AP1904  {"os":"IOS 8","model_name":"Apple Iphone5","emi_no":"APXX64314","registered_date":"07/11/2010"}
AP2345  {"os":"IOS 8","model_name":"Apple Iphone6","emi_no":"APXX64029","registered_date":"03/11/2013"}
AP8906  {"os":"IOS 7","model_name":"Apple Iphone4","emi_no":"APXX64123","registered_date":"03/11/2014"}
G671    {"os":"Android 5","model_name":"Samsung S4","emi_no":"GAXXX98765","registered_date":"12/11/2013"}
```