edX Study Lab2

(이삭엔지니어링 인턴 김형근 : hkkim@isaac-eng.com)

Lab 2 – Processing Big Data with Hive

(문서 내용 중 옥색으로 칠한 부분은, 기존의 문서에서 CDH 환경에 맞도록 내용이 변경된 부분이다. CDH 5.15에서 적용하였음.)

Overview

In this lab, you will process data in web server log files by creating Hive tables, populating them with data, and querying them.

View Source Data

1. In the **HDILabs** folder where you extracted the lab files, in the **Lab02\iislogs** folder, open any of the text files in a text editor. Each file in this folder is an Internet Information Services (IIS) log file from a web server, and there is a file for each month between January and June 2008. When you have viewed the information in the file, close it without saving any changes.
2. View the contents of the **Lab02\iislogs_gz** folder. This contains the same log files compressed using the gzip compression algorithm, significantly reducing the size of the files.

Upload the Log Files to Hue

1. Connect to 10.100.3.211:8888
2. Log in Hue (user : hkkim)
3. Set path /user/hkkim
4. Make new directory named **data**
5. In **/user/hkkim/data** directory, make directory **logs**
6. In **/user/hkkim/data/logs** directory, upload 6 logs in **Lab02\iislogs**

View the Uploaded Data Files

Now that you have a remote command console for your cluster, you can verify that the log files have been uploaded to the shared cluster storage.

1. In the **Putty**, connect to 10.100.3.211 and enter the following command to view the contents of the **/data/logs** folder in the HDFS file system.

   ```
   hdfs dfs -ls /user/hkkim/data/logs
   ```

2. Verify that the folder contains six log files.

Create a Hive Table for the Raw Log Data

While there are many Hive editors available, and you can use any one that you prefer. The instructions here assume that you will use the Hive command line interface to work with Hive in this lab.

1. In the Hue, click▽ beside **Query** and set editor to **Hive** and set database to **edx_hkkim**

2. In the Hive command line interface, enter the following HiveQL statement to create a table named **rawlogs** (you can copy and paste this from **Create Raw Table.txt** in the **HDILabs\Lab02** folder):

   ```
   CREATE TABLE rawlog
   (log_date STRING,
   log_time STRING,  c_ip
   STRING,  cs_username
   STRING,  s_ip STRING,
   s_port STRING,
   cs_method STRING,
   cs_uri_stem STRING,
   cs_uri_query STRING,
   sc_status STRING,
   sc_bytes INT,
   cs_bytes INT,
   time_taken INT,
   cs_user_agent STRING,
   cs_referrer STRING)
   ROW FORMAT DELIMITED FIELDS TERMINATED BY ' ';
   ```

3. Enter the following command to query the table, and verify that no rows are returned:

   ```
   SELECT * FROM rawlog;
   ```

Load the Source Data into the Raw Log Table

1. In the Hue interface, enter the following HiveQL statement to move the log files you previously uploaded into the folder for the **rawlogs** table (you can copy and paste this from **Load Raw Data.txt** in the **HDILabs\Lab02** folder):

```
LOAD DATA INPATH '/user/hkkim/data/logs' INTO TABLE rawlog;
```

2. Wait for the job to complete.
3. Enter the following command to return the first 100 rows in the **rawlogs** table:

```
SELECT * FROM rawlog LIMIT 100;
```

4. View the output, noting that the query retrieved the first 100 rows from the table in tab-delimited text format. Note that the output includes rows containing comments from the source document (the first column value for these rows is prefixed with a **#** character, and null values are used for columns in the table schema for which there are no values in the source data).

Clean the Log Data

1. In the Hue interface, enter the following HiveQL statement to create an external table named **cleanlog** based on the **/data/cleanlog** folder (you can copy and paste this from **Create Clean Table.txt** in the **HDILabs\Lab02** folder):

```
CREATE EXTERNAL TABLE cleanlog
(log_date DATE,
log_time STRING,  c_ip
STRING,  cs_username
STRING,  s_ip STRING,
s_port STRING,
cs_method STRING,
cs_uri_stem STRING,
cs_uri_query STRING,
sc_status STRING,
sc_bytes INT,
cs_bytes INT,
time_taken INT,
cs_user_agent STRING,
cs_referrer STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ' '
STORED AS TEXTFILE LOCATION '/user/hkkim/data/cleanlog';
```

2. Wait for the job to complete.
3. Enter the following command to extract rows that do not contain comments from the **rawlogs** table, and insert them into the **cleanlog** table (you can copy and paste this from **Load Clean Data.txt** in the **HDILabs\Lab02** folder):

```
INSERT INTO TABLE cleanlog
SELECT *
FROM rawlog
WHERE SUBSTR(log_date, 1, 1) <> '#';
```

4. Wait for the job to complete.

5. Enter the following command to retrieve the first 100 rows from the **cleanlog** table:

```
SELECT * FROM cleanlog LIMIT 100;
```

6. View the results returned by the query, noting that the **cleanlog** table does not include any rows prefixed with a **#** character.

Create a View

1. In the Hive command line interface, enter the following command to create a view named **vDailySummary** (you can copy and paste this from **Create View.txt** in the **HDILabs\Lab02** folder):

```
CREATE VIEW vDailySummary
AS
SELECT log_date,
       COUNT(*) AS requests,
       SUM(cs_bytes) AS inbound_bytes,
       SUM(sc_bytes) AS outbound_bytes
FROM cleanlog
GROUP BY log_date;
```

2. Wait for the job to complete.
3. Enter the following command to query the **vDailySummary** view:

```
SELECT * FROM vDailySummary ORDER
BY log_date;
```

4. Wait for the job to complete and then view the output returned by the query.

Partitioning Data

Partitioning data can improve performance when queries commonly filter on specific columns, or when the job can be effectively split across multiple reducer nodes.

Create and Load a Partitioned Table

1. In the Hue interface, enter the following command to create a partitioned table (you can copy and paste this from **Create Partitioned Table.txt** in the **HDILabs\Lab02** folder):

```
CREATE EXTERNAL TABLE partitionedlog
(log_day int,
log_time STRING,  c_ip
STRING,  cs_username
STRING,  s_ip STRING,
s_port STRING,
cs_method STRING,
cs_uri_stem STRING,
cs_uri_query STRING,
sc_status STRING,
sc_bytes INT,
cs_bytes INT,
time_taken INT,
cs_user_agent STRING,
cs_referrer STRING)
PARTITIONED BY (log_year int, log_month int)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ' '
STORED AS TEXTFILE LOCATION '/user/hkkim/data/partitionedlog';
```

2. Enter the following command to load data from the **rawlog** table into the partitioned table (you can copy and paste this from **Load Partitioned Table.txt** in the **HDILabs\Lab02** folder):

```
SET hive.exec.dynamic.partition.mode=nonstrict;
SET hive.exec.dynamic.partition = true;
```

```
INSERT INTO TABLE
partitionedlog
PARTITION(log_year,
log_month)
SELECT DAY(log_date),
log_time,
c_ip,
cs_username,
s_ip,        s_port,
cs_method,
cs_uri_stem,
cs_uri_query,
sc_status,
sc_bytes,
cs_bytes,
time_taken,
cs_user_agent,
cs_referrer,
YEAR(log_date),
       MONTH(log_date)
FROM rawlog
WHERE SUBSTR(log_date, 1, 1) <> '#';
```

Wait for the query job complete. By partitioning the data in this way, queries filtered on year and month will benefit from having to search a smaller volume of data to retrieve the results.

3. Enter the following command to query the partitioned table, and when the job has completed, view the results:

```
SELECT log_day, count(*) AS page_hits
FROM partitionedlog
WHERE log_year=2008 AND log_month=6
GROUP BY log_day;
```

View Folders for a Partitioned Table

1. In the **Putty**, enter the following command to exit Hive:(Connect 10.100.3.211, user : hkkim)

2. At the command prompt, enter the following command to view the **/data/partitionedlog** folder:

```
hdfs dfs -ls /user/hkkim/data/partitionedlog
```

3. Note that the folder contains a subfolder for each value in the first partitioning key (in this case, a single folder named **log_year=2008**).
4. Enter the following command to view the contents of the **log_year=2008** subfolder (note that quotation marks are required because the folder name includes a "=" character):

```
hdfs dfs -ls "/user/hkkim/data/partitionedlog/log_year=2008"
```

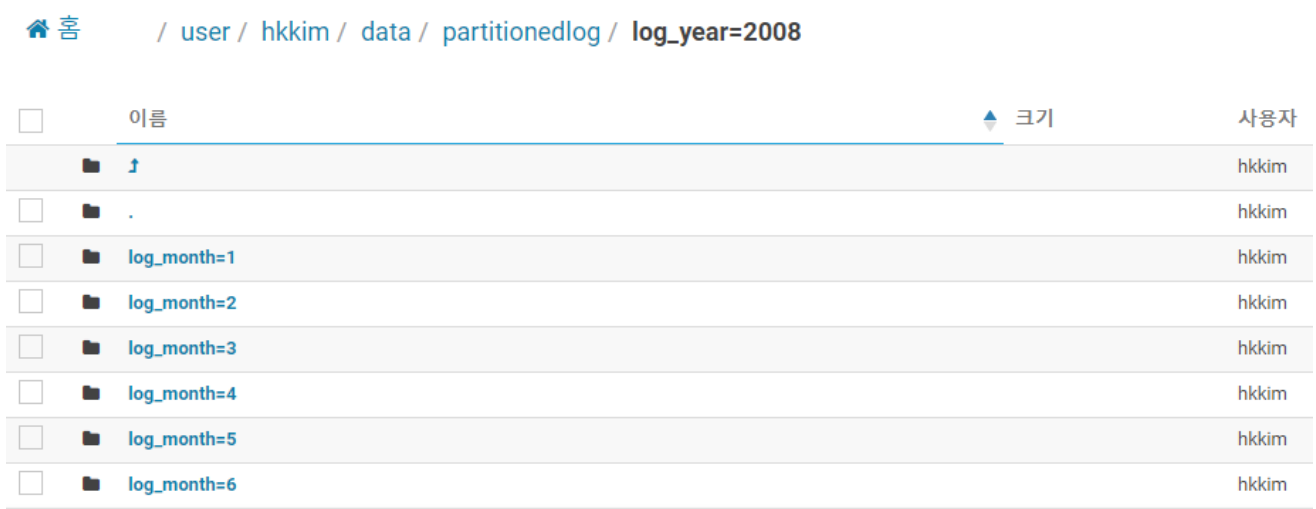
5. Note that the subfolder contains a further subfolder for each value in the second partitioning key (in this case, a folder for each **log_month** value in the source data).
6. Enter the following command to view the contents of the **log_month=6** subfolder:

```
hdfs dfs -ls "/user/hkkim/data/partitionedlog/log_year=2008/log_month=6"
```

1. Note that the subfolders for each month contain one or more data files. In this case, a file with a name similar to **000000_0** contains all of the log records with a year of 2008 and a month of 6.

2. Enter the following command to view the last few kilobytes of data in the file (modifying the file name if necessary):

```
hdfs dfs -tail "/user/hkkim
/data/partitionedlog/log_year=2008/log_month=6/000000_0"
```



3. Note that the first field in the data file contains the day of the month (the last few rows will contain the value **30**.) The year and month are not stored in the data file, but are determined from the partitioning subfolder in which the file is stored - this file contains only the records for June 2008.

   Reference : https://github.com/MicrosoftLearning/Processing-Big-Data-with-Hadoop-in-Azure-HDInsight/raw/master/Labs/Lab%202%20-%20Processing%20Big%20Data%20with%20Hive.pdf