Apache Hadoop –

A course for undergraduates

Homework Labs, Lecture 11

Lab: Running Hive Queries from the Shell, Scripts, and Hue

In this lab you will write HiveQL queries to analyze data in Hive tables that have been populated with data you placed in HDFS during earlier labs.

IMPORTANT: Since this lab builds on the previous one, it is important that you successfully complete the previous lab before starting this lab.

Step #1: Running a Query from the Hive Shell

Dualcore ran a contest in which customers posted videos of interesting ways to use their new tablets. A $5,000 prize will be awarded to the customer whose video received the highest rating.

However, the registration data was lost due to an RDBMS crash, and the only information they have is from the videos. The winning customer introduced herself only as "Bridget from Kansas City" in her video.

You will need to run a Hive query that identifies the winner's record in the customer database so that Dualcore can send her the $5,000 prize.

1. Change to the directory for this lab:

```
$ cd $ADIR/exercises/analyzing_sales
```

2. Start Hive:

```
$ hive
```

Hive Prompt

To make it easier to copy queries and paste them into your terminal window, we do not show the **`hive>`** prompt in subsequent steps. Steps prefixed with `$` should be executed on the UNIX command line; the rest should be run in Hive unless otherwise noted.

3. Make the query results easier to read by setting the property that will make Hive show column headers:

```
set hive.cli.print.header=true;
```

4. All you know about the winner is that her name is Bridget and she lives in Kansas City. Use Hive's LIKE operator to do a wildcard search for names such as "Bridget", "Bridgette" or "Bridgitte". Remember to filter on the customer's city.

   Question: Which customer did your query identify as the winner of the $5,000 prize?

Step #2: Running a Query Directly from the Command Line

You will now run a top-N query to identify the three most expensive products that Dualcore currently offers.

5. Exit the Hive shell and return to the command line:

```
quit;
```

6. Although HiveQL statements are terminated by semicolons in the Hive shell, it is not necessary to do this when running a single query from the command line using the `-e` option. Run the following command to execute the quoted HiveQL statement:

```
$ hive -e 'SELECT price, brand, name FROM PRODUCTS
ORDER BY price DESC LIMIT 3'
```

   Question: Which three products are the most expensive?

Step #3: Running a HiveQL Script

The rules for the contest described earlier require that the winner bought the advertised tablet from Dualcore between May 1, 2013 and May 31, 2013. Before

Dualcore can authorize the accounting department to pay the $5,000 prize, you must ensure that Bridget is eligible. Since this query involves joining data from several tables, it's a perfect case for running it as a Hive script.

1. Study the HiveQL code for the query to learn how it works:

```
$ cat verify_tablet_order.hql
```

2. Execute the HiveQL script using the hive command's `-f` option:

```
$ hive -f verify_tablet_order.hql
```

Question: Did Bridget order the advertised tablet in May?

Step #4: Running a Query Through Hue and Beeswax

Another way to run Hive queries is through your Web browser using Hue's Beeswax application. This is especially convenient if you use more than one computer – or if you use a device (such as a tablet) that isn't capable of running Hive itself – because it does not require any software other than a browser.

1. Start the Firefox Web browser by clicking the orange and blue icon near the top of the VM window, just to the right of the System menu. Once Firefox starts, type `http://localhost:8888/` into the address bar, and then hit the enter key.

2. After a few seconds, you should see Hue's login screen. Enter `training` in both the username and password fields, and then click the "Sign In" button. If prompted to remember the password, decline by hitting the ESC key so you can practice this step again later if you choose.

   Although several Hue applications are available through the icons at the top of the page, the Beeswax query editor is shown by default.

3. Select `default` from the database list on the left side of the page.

4. Write a query in the text area that will count the number of records in the `customers` table, and then click the "Execute" button.

   Question: How many customers does Dualcore serve?

5. Click the "Query Editor" link in the upper left corner, and then write and run a query to find the ten states with the most customers.

   Question: Which state has the most customers?

Bonus Lab #1: Calculating Revenue and Profit

Several more questions are described below and you will need to write the HiveQL code to answer them. You can use whichever method you like best, including Hive shell, Hive Script, or Hue, to run your queries.

- Which top three products has Dualcore sold more of than any other?
  Hint: Remember that if you use a `GROUP BY` clause in Hive, you must group by all fields listed in the `SELECT` clause that are not part of an aggregate function.
- What was Dualcore's total revenue in May, 2013?
- What was Dualcore's gross profit (sales price minus cost) in May, 2013?
- The results of the above queries are shown in cents. Rewrite the gross profit query to format the value in dollars and cents (e.g., $2000000.00). To do this, you can divide the profit by 100 and format the result using the `PRINTF` function and the format string `"$%.2f"`.

This is the end of the lab.

Lab: Data Management with Hive

In this lab you will practice using several common techniques for creating and populating Hive tables. You will also create and query a table containing each of the complex field types we studied: array, map, and struct.

IMPORTANT: Since this lab builds on the previous one, it is important that you successfully complete the previous lab before starting this lab.

Additionally, many of the commands you will run use environmental variables and relative file paths. It is important that you use the Hive shell, rather than Hue or another interface, as you work through the steps that follow.

Step #1: Use Sqoop's Hive Import Option to Create a Table

You used Sqoop in an earlier lab to import data from MySQL into HDFS. Sqoop can also create a Hive table with the same fields as the source table in addition to importing the records, which saves you from having to write a `CREATE TABLE` statement.

1. Change to the directory for this lab:

```
$ cd $ADIR/exercises/data_mgmt
```

2. Execute the following command to import the `suppliers` table from MySQL as a new Hive-managed table:

```
$ sqoop import \
  --connect jdbc:mysql://localhost/dualcore \
  --username training --password training \
  --fields-terminated-by '\t' \
  --table suppliers \
  --hive-import
```

3. Start Hive:

```
$ hive
```

4. It is always a good idea to validate data after adding it. Execute the Hive query shown below to count the number of suppliers in Texas:

```
SELECT COUNT(*) FROM suppliers WHERE state='TX';
```

The query should show that nine records match.

Step #2: Create an External Table in Hive

You imported data from the `employees` table in MySQL in an earlier lab, but it would be convenient to be able to query this from Hive. Since the data already exists in HDFS, this is a good opportunity to use an external table.

1. Write and execute a HiveQL statement to create an external table for the tab-delimited records in HDFS at `/dualcore/employees`. The data format is shown below:

Field Name	Field Type
emp_id	STRING
fname	STRING
lname	STRING
address	STRING
city	STRING
state	STRING
zipcode	STRING
job_title	STRING
email	STRING
active	STRING
salary	INT

2. Run the following Hive query to verify that you have created the table correctly.

```
SELECT job_title, COUNT(*) AS num
    FROM employees
    GROUP BY job_title
    ORDER BY num DESC
    LIMIT 3;
```

It should show that Sales Associate, Cashier, and Assistant Manager are the three most common job titles at Dualcore.

Step #3: Create and Load a Hive-Managed Table

Next, you will create and then load a Hive-managed table with product ratings data.

1. Create a table named `ratings` for storing tab-delimited records using this structure:

Field Name	Field Type
posted	TIMESTAMP
cust_id	INT
prod_id	INT
rating	TINYINT
message	STRING

2. Show the table description and verify that its fields have the correct order, names, and types:

```
DESCRIBE ratings;
```

3. Next, open a separate terminal window (File -> Open Terminal) so you can run the following shell command. This will populate the table directly by using the `hadoop fs` command to copy product ratings data from 2012 to that directory in HDFS:

```
$ hadoop fs -put $ADIR/data/ratings_2012.txt \
/user/hive/warehouse/ratings
```

   Leave the window open afterwards so that you can easily switch between Hive and the command prompt.

4. Next, verify that Hive can read the data we just added. Run the following query in Hive to count the number of records in this table (the result should be 464):

```
SELECT COUNT(*) FROM ratings;
```

5. Another way to load data into a Hive table is through the `LOAD DATA` command. The next few commands will lead you through the process of copying a local file to HDFS and loading it into Hive. First, copy the 2013 ratings data to HDFS:

```
$ hadoop fs -put $ADIR/data/ratings_2013.txt /dualcore
```

6. Verify that the file is there:

```
$ hadoop fs -ls /dualcore/ratings_2013.txt
```

7. Use the `LOAD DATA` statement in Hive to load that file into the `ratings` table:

```
LOAD DATA INPATH '/dualcore/ratings_2013.txt' INTO
TABLE ratings;
```

8. The `LOAD DATA INPATH` command *moves* the file to the table's directory. Verify that the file is no longer present in the original directory:

```
$ hadoop fs -ls /dualcore/ratings_2013.txt
```

9. Verify that the file is shown alongside the 2012 ratings data in the table's directory:

```
$ hadoop fs -ls /user/hive/warehouse/ratings
```

10. Finally, count the records in the ratings table to ensure that all 21,997 are available:

```
SELECT COUNT(*) FROM ratings;
```

Step #4: Create, Load, and Query a Table with Complex Fields

Dualcore recently started a loyalty program to reward their best customers. Dualcore has a sample of the data that contains information about customers who have signed up for the program, including their phone numbers (as a map), a list of past order IDs (as an array), and a struct that summarizes the minimum, maximum, average, and total value of past orders. You will create the table, populate it with the provided data, and then run a few queries to practice referencing these types of fields.

1. Run the following statement in Hive to create the table:

```
CREATE TABLE loyalty_program
   (cust_id INT,
    fname STRING,
    lname STRING,
    email STRING,
    level STRING,
    phone MAP<STRING, STRING>,
    order_ids ARRAY<INT>,
    order_value STRUCT<min:INT,
                       max:INT,
                       avg:INT,
                       total:INT>)
   ROW FORMAT DELIMITED
      FIELDS TERMINATED BY '|'
      COLLECTION ITEMS TERMINATED BY ','
      MAP KEYS TERMINATED BY ':';
```

2. Examine the data in `loyalty_data.txt` to see how it corresponds to the fields in the table and then load it into Hive:

```
LOAD DATA LOCAL INPATH 'loyalty_data.txt' INTO TABLE
loyalty_program;
```

3. Run a query to select the `HOME` phone number (Hint: Map keys are case-sensitive) for customer ID 1200866. You should see 408-555-4914 as the result.

4. Select the third element from the `order_ids` array for customer ID 1200866 (Hint: Elements are indexed from zero). The query should return 5278505.

5. Select the `total` attribute from the `order_value` struct for customer ID 1200866. The query should return 401874.

Bonus Lab #1: Alter and Drop a Table

1. Use `ALTER TABLE` to rename the `level` column to `status`.

2. Use the `DESCRIBE` command on the `loyalty_program` table to verify the change.

3. Use `ALTER TABLE` to rename the entire table to `reward_program`.

4. Although the `ALTER TABLE` command often requires that we make a corresponding change to the data in HDFS, renaming a table or column does not. You can verify this by running a query on the table using the new names (the result should be "SILVER"):

```
SELECT status FROM reward_program WHERE cust_id =
1200866;
```

5. As sometimes happens in the corporate world, priorities have shifted and the program is now canceled. Drop the `reward_program` table.

This is the end of the lab.