Data Analysis using Hadoop: Module 4, Lesson 7  
Hive DDL Hands-On Lab

## Overview

In this lab, you will learn how to create internal managed tables and external tables.

## Objectives

In this hands-on lab you will learn how to:

* Create external tables
* Create internal managed tables
* Load data onto internal tables

## Prerequisites

The following are required to complete this hands-on lab:

* Completion of Module 4 Lesson 3 Lab
* A provisioned HDInsight cluster
* A web browser

Note: The Azure portal is continually improved and changed. The steps in this exercise reflect the user interface of the Microsoft Azure portal at the time of writing, but may not match the latest design of portal.

## Exercises

This hands-on lab includes the following exercises:

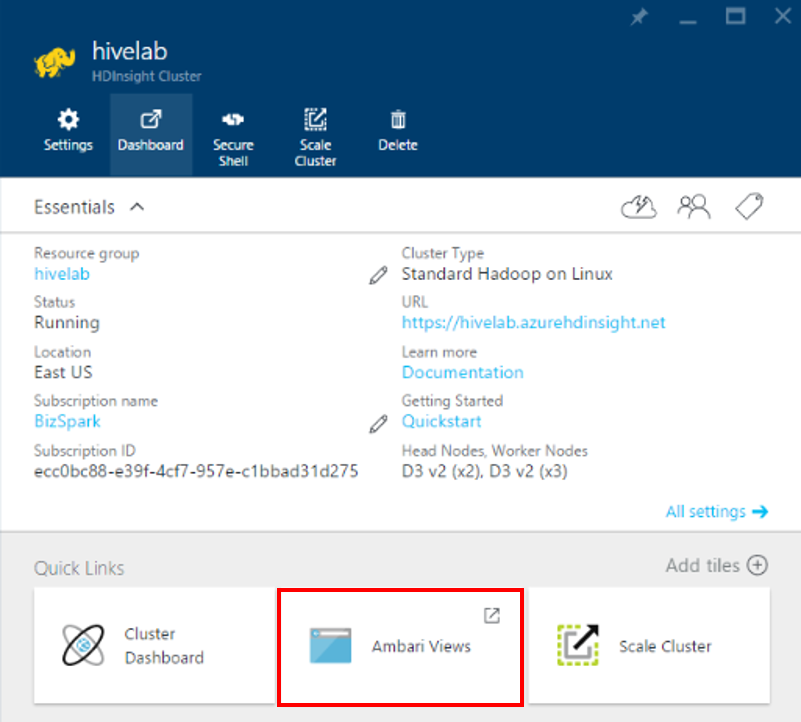
* Exercise 1: Create an external table
* Exercise 2: Create an internal managed table

## Exercise 1: Create and external table

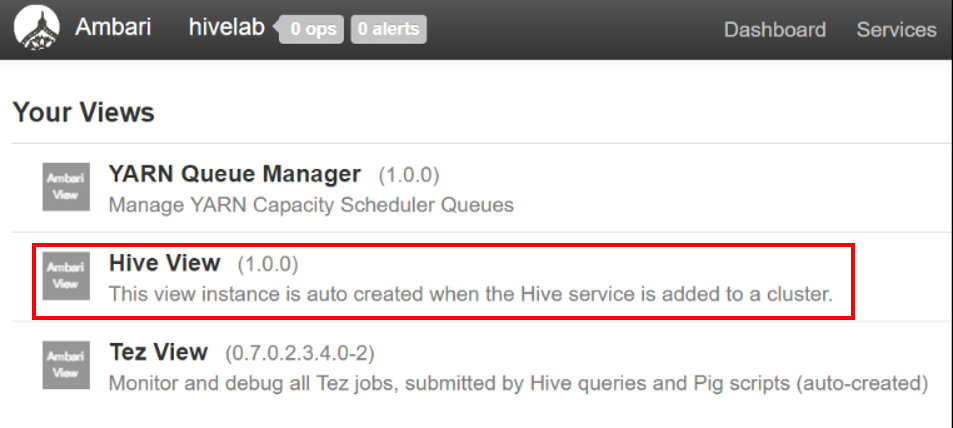
In this exercise, you will learn how to use Hive View to create an external table.

1. Access the Hive View for the hivelab HDInsight cluster

Select the hivelab HDInsight Cluster from the Azure portal



Once you have selected the hivelab HDInsight cluster, you will see the “Quick Links” section. From there, click on “Ambari Views” and then select Hive View.



1. Create external table in the query editor

In the Query Editor worksheet, enter the following and click Execute.

DROP TABLE log4jLogs;

CREATE EXTERNAL TABLE log4jLogs

(t1 string,

t2 string,

t3 string,

t4 string,

t5 string,

t6 string,

t7 string)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ' '

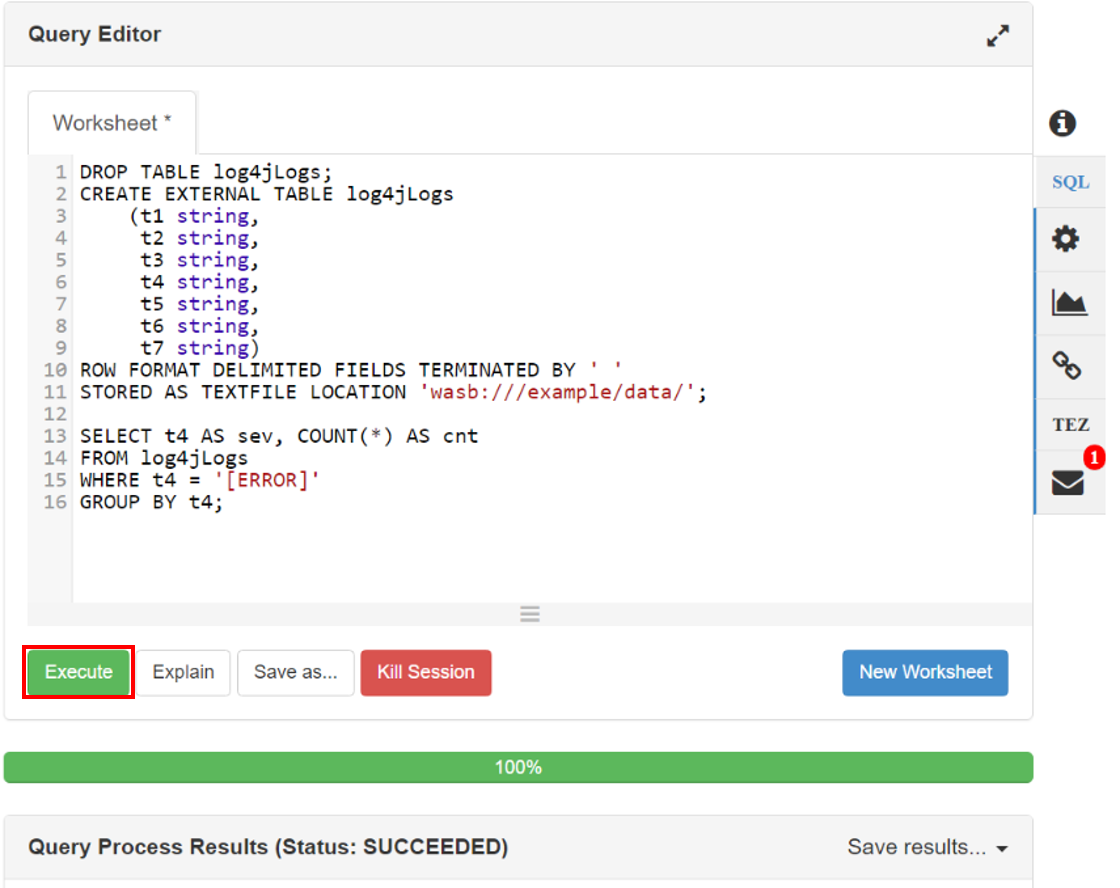
STORED AS TEXTFILE LOCATION 'wasb:///example/data/';

SELECT t4 AS sev, COUNT(\*) AS cnt

FROM log4jLogs

WHERE t4 = '[ERROR]'

GROUP BY t4;



These statements perform the following actions:

* **DROP TABLE** - Deletes the table and the data file, in case the table already exists.
* **CREATE EXTERNAL TABLE** - Creates a new "external" table in Hive. External tables store only the table definition in Hive; the data is left in the original location.
* **ROW FORMAT** - Tells Hive how the data is formatted. In this case, the fields in each log are separated by a space.
* **STORED AS TEXTFILE LOCATION** - Tells Hive where the data is stored (the example/data directory), and that it is stored as text.
* **SELECT** - Selects a count of all rows where column t4 contains the value [ERROR].

External tables should be used when you expect the underlying data to be updated by an external source, such as an automated data upload process, or by another MapReduce operation, but you always want Hive queries to use the latest data. Dropping an external table does **NOT** delete the data, only the table definition.

1. Execute the query

Use the **Execute** button at the bottom of the Query Editor to start the query. It should turn orange and the text will change to **Stop execution**. A **Query Process Results** section should appear beneath the Query Editor and display information about the job.

1. Review the results

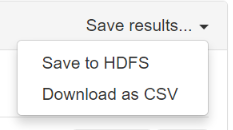
Once the query has finished, The **Query Process Results** section will display the results of the operation. The **Stop execution** button will also change back to a green **Execute** button. The **Results** tab should contain the following information:

sev cnt

[ERROR] 3

1. Save the result

You can save the results of your query. Click the **Save results** drop-down dialog in the upper right of the **Query Process Results** section; you can use this to either download the results, or save them to HDInsight storage as a CSV file.



1. Execute selection

Select the first four lines of this query, then select **Execute**. Notice that there are no results when the job completes. This is because using the **Execute** button when part of the query is selected will only run the selected statements. In this case, the selection didn't include the final statement that retrieves rows from the table. If you select just that line and use **Execute**, you should see the expected results.

## Exercise 2: Create an Internal Table

In this exercise, we will create an internal managed table from Hive View.

1. Create internal managed table

Use the **New Worksheet** button at the bottom of the **Query Editor** to create a new worksheet. In the new worksheet, enter the following HiveQL statements:

CREATE TABLE IF NOT EXISTS errorLogs

(t1 string,

t2 string,

t3 string,

t4 string,

t5 string,

t6 string,

t7 string)

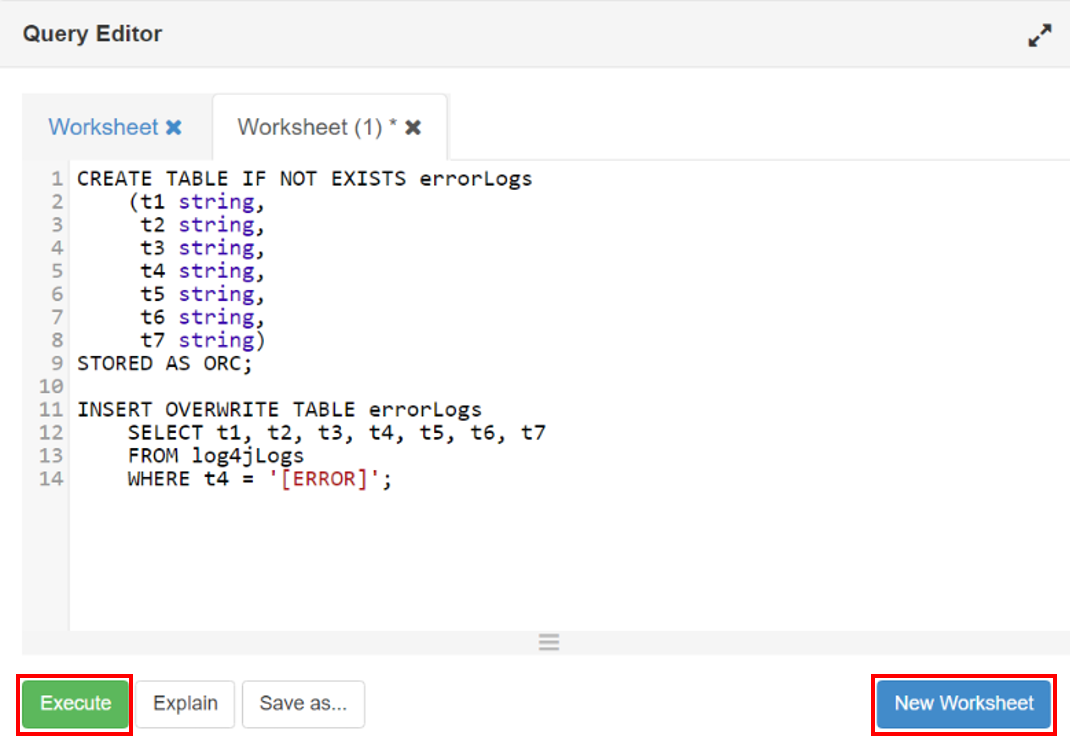
STORED AS ORC;

INSERT OVERWRITE TABLE errorLogs

SELECT t1, t2, t3, t4, t5, t6, t7

FROM log4jLogs

WHERE t4 = '[ERROR]';



These statements perform the following actions:

* **CREATE TABLE IF NOT EXISTS** - Creates a table, if it does not already exist. Since the **EXTERNAL** keyword is not used, this is an internal table, which is stored in the Hive data warehouse and is managed completely by Hive. Unlike external tables, dropping an internal table will delete the underlying data as well.
* **STORED AS ORC** - Stores the data in Optimized Row Columnar (ORC) format. This is a highly optimized and efficient format for storing Hive data.
* **INSERT OVERWRITE ... SELECT** - Selects rows from the **log4jLogs** table that contain [ERROR], and then inserts the data into the **errorLogs** table.

1. Execute the query

Use the **Execute** button to run this query. The **Results** tab will not contain any information as no rows are returned by this query, but the status should show as **SUCCEEDED**.

## Summary

In this hands-on lab, you learned how to:

* Use the Query Editor in Hive View
* Create an external table and connect an external data source to it
* Save the results of a query to HDFS or download as a CVS file
* Create an internal table and load it with data