Examining Table Structure

In real-world contexts, it’s common that *after* you create a table with Hive or Impala, you realize that the data would be better represented using a different schema, or that some other property of the table needs to be modified. So you’ll often need to make changes to tables. You'll see how to do this in the “Modifying Existing Tables” reading later in this lesson.

But before you modify a table, you’ll want to examine it to see its existing schema and other properties. There are two commands that are useful to help you understand the state of your tables: DESCRIBE and SHOW CREATE TABLE. Additionally, SHOW CREATE TABLE is useful when you need to recreate tables in a different environment.

DESCRIBE (and DESCRIBE FORMATTED)

You might recall the DESCRIBE utility statement. You can use the DESCRIBE statement to see what columns are in a table, by running the command DESCRIBE *tablename*. The results show the names and data types of all the columns, and sometimes a comment for each column.

To see more detailed information about a table, use the DESCRIBE FORMATTED command. That command shows additional details, including the file format and storage location of the table’s data files.

SHOW CREATE TABLE

Another way to understand the structure and properties of a table is to see the CREATE TABLE statement that created it. With Hive and Impala, you can do this using the SHOW CREATE TABLE statement.

Furthermore, if you made changes to the schema or other properties of a table after creating it, then the output of the SHOW CREATE TABLEstatement will reflect all of those changes. This makes it particularly useful for recreating a table; instead of issuing the original CREATE TABLE statement, followed by a series of other statements to modify the table, you can use SHOW CREATE TABLE to display a single CREATE TABLE statement to recreate the table in its current state. You can copy that CREATE TABLE statement and execute it in a different environment that doesn't share the same metastore. This is especially useful when migrating tables from a development or test environment to a production environment.

Try It!

First compare the results of a DESCRIBE statement with and without the FORMATTED keyword.

1. Execute the following commands in Hive and notice the difference in the details provided. (You must use Hive because the dig.tunnels table was created using a Hive SerDe.)

```
DESCRIBE dig.tunnels;

DESCRIBE FORMATTED dig.tunnels;
```

Do Steps 2 and 3 to see how you can tell if a table is (internally) managed or unmanaged (that is, externally managed).

2. Look again at the DESCRIBE FORMATTED results for dig.tunnels. Look down the results for Table Type; the value should be MANAGED_TABLE. This means it was created *without* the EXTERNAL keyword.

3. Compare that to an externally managed table. You can run this in Hive or Impala:

```
DESCRIBE FORMATTED default.investors;
```

Again look for Table Type and note the value for it.

4. The dig.tunnels table was created with a SerDe. Look again at the results of the DESCRIBE FORMATTED command for that table, or re-run the command in Hive if necessary. Look for SerDe Library in the col_name column, and see what the data_type value is for that.

5. Although you haven't made any modifications to a table, try the SHOW CREATE TABLE statement with the default.investors table:

```
SHOW CREATE TABLE default.investors;
```

Take some time to review the result. You'll see a lot of familiar things (like EXTERNAL and ROWS DELIMITED keywords); but you'll probably see some things that are not so familiar. For this table, you'll see TBLPROPERTIES settings that you don't recall setting and you might not understand. These are settings that happen invisibly, by default. Don't worry about it—this is not the statement you should have used to create the table; instead, it's *one possible* statement that you *can* use to produce the exact table that you currently have.