Querying Complex Data with Hive

A Hive query can select a full complex column simply by including the bare name of the column in the SELECT list. For example, this query selects the full ARRAY column named phones. Hive displays the full phones column in the results, using square brackets and commas to represent the ARRAY structure:

```
SELECT name, phones FROM customers_phones_array;
```

name	phones
Alice	[555-1111, 555-2222, 555-3333]
Bob	[555-4444]
Carlos	[555-5555, 555-6666]

For access to the elements within the different complex data types, you have to use different syntax depending on the complex type.

Note: The syntax in this reading is for Hive only. See “Querying Complex Data with Impala” for the syntax to use with Impala.

Querying ARRAYs with Hive

To query an element within an ARRAY, use an array index number in square brackets. The array index starts at 0. For example, to get the first and second phone numbers in phones, use this query:

```
SELECT name, phones[0], phones[1]
    FROM customers_phones_array;
```

Since Bob has only one phone number, the query returns NULL for Bob’s second phone number.

Querying MAPs with Hive

Querying an element within a MAP is similar to querying the ARRAY element, except you use the key instead of the index. For example, to get the home phone numbers, use this query:

```
SELECT name, phones['home'] AS home
    FROM customers_phones_map;
```

In this example, the MAP keys are strings, so you must quote the literal string within the square brackets. MAP keys are case-sensitive, so 'HOME' or 'Home' would not work in this case.

Since Bob has only a mobile phone number, the query returns NULL for Bob’s home phone number.

Querying STRUCTs with Hive

To query a field from a STRUCT column, use the column name, a dot, and the field name (similar to how you can use the database name, a dot, and the table name to refer to a table in a different database from the active one). For example, this query selects the name column, and the state and zipcode fields from the address column:

```
SELECT name, address.state, address.zipcode
    FROM customers_addr;
```

In this example, Carlos's address is missing the state and zipcode fields, so the query returns NULL for these missing fields.

Try It!

Do the following to run two queries on each of the three (non-Parquet) tables you created in "Creating Tables with Hive and Impala." Use Hive for these exercises.

1. First, run SELECT * FROM *tablename*; for each table and note how the complex column appears in the results.
2. Then, run each of the examples in the reading above. Notice the NULL fields in each case.
3. *Optional:* Try some other queries for each table.