CassandraCli06

Cassandra ships with a very basic interactive command line interface, or shell. Using the CLI you can connect to remote nodes in the cluster, set and retrieve records and columns, or query node and cluster meta-data (i.e. cluster name, keyspace listings and disposition, etc). The CLI is handy for quick tests or for familiarizing yourself with the data-model.

You can start the CLI using the `bin/cassandra-cli` startup script.

```
evans@achilles:~/cassandra$ bin/cassandra-cli -host localhost -port 9160
Connected to localhost/9160
Welcome to cassandra CLI.

Type 'help' or '?' for help. Type 'quit' or 'exit' to quit.
cassandra>
```

If you are using SimpleAuthenticator (instead of AllowAllAuthenticator) you can specify the username, password and keyspace by adding the following options to cassandra-cli: -username username -password password -keyspace keyspace):

```
tblose@quasar:~/dev/workspaces/cassandra$ bin/cassandra-cli -host localhost -port 9160 -username todd -keyspace
Keyspace1 -password blah
Connected to: "Test Cluster" on localhost/9160
Welcome to cassandra CLI.

Type 'help' or '?' for help. Type 'quit' or 'exit' to quit.
[todd@Keyspace1]
```

As the banner says, you can use 'help' or '?' to see what the CLI has to offer, and 'quit' or 'exit' when you've had enough fun. But lets try something slightly more interesting...

```
[todd@Keyspace1] set Keyspace1.Standard2['jsmith']['first'] = 'John'
Value inserted.
[todd@Keyspace1] set Keyspace1.Standard2['jsmith']['last'] = 'Smith'
Value inserted.
[todd@Keyspace1] set Keyspace1.Standard2['jsmith']['age'] = '42'
Value inserted.
```

Cassandra 0.7 introduces the `use` command to reduce keystrokes. The above example can be reduced to...

```
[default@unknown] use Keyspace1 todd 'blah$'
Authenticated to keyspace: Keyspace1
[todd@Keyspace1] set Standard2['jsmith']['first'] = 'John'
Value inserted.
[todd@Keyspace1] set Standard2['jsmith']['last'] = 'Smith'
Value inserted.
[todd@Keyspace1] set Standard2['jsmith']['age'] = '42'
Value inserted.
```

In the example above we authenticated to 'Keyspace1' and created a record in the `Standard2` column family using the key `jsmith`. This record has three columns, `first`, `last`, and `age`. Each of these commands is the equivalent to an `insert()` using the Thrift API.

In API version 2.1.0 the example would go like the following due to changes in the API.

```
cassandra> set Keyspace1.Standard2['jsmith']['first'] = 'John'
Value inserted.
cassandra> set Keyspace1.Standard2['jsmith']['last'] = 'Smith'
Value inserted.
cassandra> set Keyspace1.Standard2['jsmith']['age'] = '42'
Value inserted.
```

Now let's read back the `jsmith` row to see what it contains:

```
[todd@Keyspace1] get Standard2['jsmith']
=> (column=last, value=Smith, timestamp=1271921526614000)
=> (column=first, value=John, timestamp=1271921521923000)
=> (column=age, value=42, timestamp=1271921532713000)
Returned 3 results.
```

Note: Using the `get` command in this form is the equivalent to a `get_slice()` using the Thrift API.

Once again, with the API being different, you need to specify the Keyspace

```
cassandra> get Keyspace1.Standard2['jsmith']
=> (column=last, value=Smith, timestamp=1278953905903000)
=> (column=first, value=John, timestamp=1278953848952000)
=> (column=age, value=42, timestamp=1278953919182000)
Returned 3 results.
```