04_Drivers

READY

READY

```
%sh
STATUS="$(service cassandra status)"

if [[ $STATUS == *"is running"* ]]; then
    echo "Cassandra is running"
else
    echo " Cassandra not running .... Starting"
    service cassandra restart > /dev/null 2>&1 &
    echo " Started"
fi
```

READY

Exercise 4 – Drivers

In this exercise, you will:

- You will understand what Apache Cassandra™ drivers are and their purpose.
- You will be able to create and read records using a driver.

As we have already seen, we can access Apache Cassandra™ for client applications such as CQLSH. However, we may want to access Apache Cassandra™ directly from within an application
we create. Drivers are the mechanisms we use to interact with Apache Cassandra™ from a programming language. In this exercise, we will connect to our Apache Cassandra™ database
using the Python driver and read and write some data.

READY

Steps

1. Back in your Terminal window, make sure Apache Cassandra is still running with nodetool status. If not, restart Apache Cassandra.

READY

2. Make sure Python interpreter is installed:

READY

```
%sh
python --version
```

READY

3. Let's write Python code to connect to the cluster and open a session:

READY

04_Drivers

```
%python

from cassandra.cluster import Cluster
cluster = Cluster(protocol_version = 3)
session = cluster.connect('yootoob')
```

READY

NOTE: If you have trouble connecting, your python driver may be out of date. Use the following command to update the driver.

READY

```
%sh
pip install --upgrade cassandra-driver
```

READY

4. Now write code to retrieve all records in the videos_by_tag table. Display your results using Python's print() function:

READY

READY

5. session.execute() returns a sequence of rows (tuples), which you can further index into to get cell values. Try executing the following:

READY

```
print('{0:12} {1:40} {2:5}'.format('Tag', 'ID', 'Title'))
for val in session.execute("select * from videos_by_tag"):
    print('{0:12} {1:40} {2:5}'.format(val[0], str(val[2]), val[3]))
```

READY

We use some simple string formatting to make the output bearable. Notice we index into each tuple using the bracket operator.

READY

6. Write some Python code to insert a new video into the database:

READY

READY

7. Now run the following code to view your new record:

READY

04_Drivers

READY

```
print('{0:12} {1:40} {2:5}'.format('Tag', 'ID', 'Title'))
for val in session.execute("select * from videos_by_tag"):
    print('{0:12} {1:40} {2:5}'.format(val[0], str(val[2]), val[3]))
```

8. Now write Python code to delete your record:

READY

READY

Note to instructor - be sure to use the correct UUID for video_id.

9. Now execute the following code to confirm that your record is gone:

READY

```
print('{0:12} {1:40} {2:5}'.format('Tag', 'ID', 'Title'))
for val in session.execute("select * from videos_by_tag"):
    print('{0:12} {1:40} {2:5}'.format(val[0], str(val[2]), val[3]))
```

READY

READY

READY