# 01_CQL_Solution

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 1 – CQL

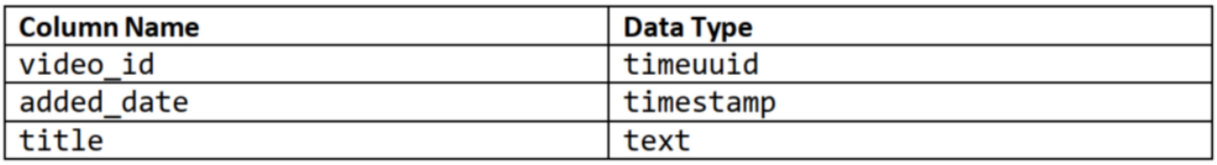
In this exercise, you will:

- Create a keyspace for YooToob
- Create a table to store video metadata
- Load the data for the video table from a CSV file

# Steps

Welcome to the YooToob company! YooToob hired you to build the latest and greatest video sharing application on the Internet. Your task is to ramp up on the domain and become acquainted with Apache Cassandra™. To start, you decided to look into creating a table schema
and to load some video data.

The video metadata is made up of:



| Column Name | Data Type |
|---|---|
| video_id | timeuuid |
| added_date | timestamp |
| title | text |

READY

1. Make sure Apache Cassandra is still running with nodetool status. If not, restart Apache Cassandra.

```
%sh
nodetool status
```

READY

2. In the terminal window, verify cqlsh is working:

READY

```
%sh
cqlsh
```

READY

3. In cqlsh, create a keyspace called YooToob. Use SimpleStrategy for the replication class with a replication factor of one.

READY

```
%cassandra

CREATE KEYSPACE YooToob
WITH replication = {
'class':'SimpleStrategy',
'replication_factor': 1
};
```

READY

4. In cqlsh switch to the newly created keyspace with the USE command.

READY

```
USE YooToob;
```

READY

5. Create a single table called videos with the same structure as shown above. **video_id** is the primary key.

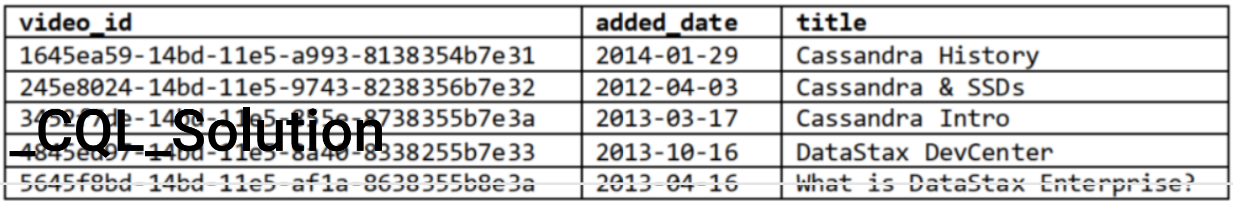
READY

```
CREATE TABLE videos (
video_id TIMEUUID,
added_date TIMESTAMP,
title TEXT,
PRIMARY KEY (video_id)
);
```

READY

6. Manually insert a single record into the table using INSERT command. Use the first row from the table below:

READY



| video_id | added_date | title |
|---|---|---|
| 1645ea59-14bd-11e5-a993-8138354b7e31 | 2014-01-29 | Cassandra History |
| 245e8024-14bd-11e5-9743-8238356b7e32 | 2012-04-03 | Cassandra & SSDs |
| 3452f7de-14bd-11e5-855e-8738355b7e3a | 2013-03-17 | Cassandra Intro |
| 4845ed97-14bd-11e5-8a40-8338255b7e33 | 2013-10-16 | DataStax DevCenter |
| 5645f8bd-14bd-11e5-af1a-8638355b8e3a | 2013-04-16 | What is DataStax Enterprise? |

```
INSERT INTO videos (video_id, added_date, title)
VALUES (1645ea59-14bd-11e5-a993-8138354b7e31, '2014-01-29', 'Cassandra History');
```

READY

7. Write a select statement to verify your record was inserted.

READY

```
SELECT * FROM videos;
```

READY

8. Insert the second record as well and run a select statement to verify it’s there.

READY

**NOTE:** You should now see two records in your videos table.

```
INSERT INTO videos (video_id, added_date, title)
VALUES (245e8024-14bd-11e5-9743-8238356b7e32, '2012-04-03', 'Cassandra & SSDs');
```

READY

9. Let’s remove the data you inserted using the TRUNCATE command.

READY

```
TRUNCATE videos;
```

READY

10. Execute the following command to import data into your videos table.

READY

```
INSERT INTO videos (video_id, added_date, title)
VALUES (1645ea59-14bd-11e5-a993-8138354b7e31, '2014-01-29', 'Cassandra History');

INSERT INTO videos (video_id, added_date, title)
VALUES (245e8024-14bd-11e5-9743-8238356b7e32, '2012-04-03', 'Cassandra & SSDs');

INSERT INTO videos (video_id, added_date, title)
VALUES (3452f7de-14bd-11e5-855e-8738355b7e3a, '2013-03-17', 'Cassandra Intro');

INSERT INTO videos (video_id, added_date, title)
VALUES (4845ed97-14bd-11e5-8a40-8338255b7e33, '2013-10-16', 'Apache Cassandra');

INSERT INTO videos (video_id, added_date, title)
VALUES (5645f8bd-14bd-11e5-af1a-8638355b8e3a, '2013-04-16', 'What is Apache Cassandra?');
```

READY

11. Use SELECT to verify the data loaded correctly.

READY

```
SELECT *
FROM videos;
```

READY

12. Use SELECT to COUNT(*) the number of imported rows. It should match the number of rows COPY reported as imported.

READY

```
SELECT COUNT(*)
FROM videos;
```

READY

READY