READY

02_Partitions_Solution

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 2 – Partitions

In this exercise, you will:

- Experiment with partitions

READY

1. NOTE: Be sure Apache Cassandra™ is running before doing these exercises. You can check by
running nodetool on the command line:

READY

```
%sh
nodetool status
```

READY

2. Switch to the YooToob keyspace via the USE command:

READY

```
%cassandra

USE YooToob;
```

READY

3. Execute the following command to view the metadata for the videos table you created earlier.

```
DESCRIBE TABLE videos;
```

READY

- What is the partition key?
- How many partitions are in this table?

READY

Answers:

READY

video_id

One for each unique primary key value.

4. Execute the following query to view the partitioner token value for each video id.

READY

```
SELECT token(video_id), video_id
FROM videos;
```

READY

5. Use the following command to inspect the file named /home/labwork/data-files/videos-by-tag.csv:

READY

```
%sh
cat /home/labwork/data-files/videos-by-tag.csv
```

READY

6. Switch to the YooToob keyspace.

READY

```
USE YooToob;
```

READY

7. Your mission, should you choose to accept it, is to write a CREATE TABLE statement that will store this data partitioned by tags. With this given data set, there should be two partitions, one for each tag. Call your table videos_by_tag.

READY

```
CREATE TABLE videos_by_tag (
tag TEXT,
video_id UUID,
added_date TIMESTAMP,
title TEXT,
PRIMARY KEY ((tag), video_id)
);
```

READY

8. Execute the following COPY command to import the videos-by-tag.csv data.

READY

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

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

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

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

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

READY

9. Verify CQL imported your data correctly by writing a SELECT * command.

READY

```
SELECT *
FROM videos_by_tag;
```

READY

10. Write a SELECT statement to retrieve all rows tagged with cassandra.

READY

```
SELECT *
FROM videos_by_tag
WHERE tag = 'cassandra';
```

READY

11. Now, find all videos tagged with fenago (similar to the previous query).

READY

```
FROM videos_by_tag
WHERE tag = 'fenago';
```

READY

12. Finally, write a query to retrieve the video having a title of Cassandra Intro.

READY

```
SELECT *
FROM videos_by_tag
WHERE title = 'Cassandra Intro';
```

READY

READY

NOTE: Notice your query errors out. Apache Cassandra™ only allows queries on the partition key (and clustering columns shown in the next section). Since `title` is not the partition key, Apache Cassandra™ fails the query. If Apache Cassandra™ allowed querying on non-partition key columns, Apache Cassandra™ would have to scan all partitions on all nodes to produce a result set (which goes against the reason you would use Apache Cassandra™ in the first place).

READY

READY