Exercise 3.2: Using Counters in CQL

In this exercise, you will:

- Create a new table that makes use of the counter type
- Load the newly created table with data
- Run queries against the table to test counter functionality

Background

You have been on a roll lately designing tables, and that hasn't gone unnoticed. Since no good deed goes unpunished, your team has unanimously agreed to let you tackle one of the trickier problems they encountered learning Cassandra.

The problem is with keeping track of the number of videos in a particular category. Whenever a tag is added to a video, a transaction is done which reads the number of videos for that video category, increments it by one, and updates it in the database. However, this technique is susceptible to race conditions. Cassandra is a distributed database that allows for concurrent, simultaneous operations, and it does not allow transactions. Your team has been stumped on how to get this functionality working correctly in Cassandra. Fortunately, you have just read up on the counter type and know that it is the perfect way to solve this problem.

The requirement itself is to be able to support a query that can retrieve the number of videos for a particular category, defined as a specific tag and year added. The query allows querying on a tag, and optionally, on a range for the added year.

Steps

1. Navigate to '/home/ubuntu/labwork/counters' and open the 'videos_count_by_tag.cql' file and review its contents. Notice this is a CQL script rather than a CSV data file.

2. Launch 'cqlsh' and switch to 'killrvideo' keyspace.

3. Create a new table called 'videos_count_by_tag' with a column 'video_count' which makes use of a counter type to store the video count. Structure your table to work correctly with the CQL in 'videos_count_by_tag.cql'.

4. Load the number of counts from the 'videos_count_by_tag.cql' file into the 'videos_count_by_tag' table using the SOURCE command.

5. Run a query to display each category of tag and added year, along with the count of videos for each.

6. Simulate adding another a tag for another video by incrementing the video count for a category, and then querying the new count from the 'videos_count_by_tag' table.

7. Exit cqlsh.