Exercise 17 – Read Path

In this exercise, you will:

- Understand the Apache Cassandra read path.

Steps

1) To begin, we need to populate our single-node cluster with a good chunk of data. We will use `cassandra-stress` to do so. Execute the following command in a terminal window. Wait for `cassandra-stress` to complete before continuing.

   ```
   /home/ubuntu/node/resources/cassandra/tools/bin/cassandra-stress
   write no-warmup n=250000 -port native=9041 -rate threads=1
   ```

   Take a visit to the water cooler while this executes. Catch up with your coworkers, shoot the bull, etc.

2) Now force Apache Cassandra™ to flush its current memtable to disk by executing the following command:

   ```
   /home/ubuntu/node/resources/cassandra/bin/nodetool flush
   ```

3) In your terminal, navigate to the data directory for the large table that `cassandra-stress` wrote. Remember that `cassandra-stress` creates a keyspace called `keyspace1` and a table called `standard1`. You will find the directory as illustrated below, however, your table ID will differ. Use the tab key in the terminal to use helpful autocompletion.

   ```
   ubuntu@ds201-node1:~$ cd
   /home/ubuntu/node/data/data/keyspace1/standard1-9b3c...0e72657334
   ubuntu@ds201-
   node1:/home/ubuntu/node/data/data/keyspace1/standard1-
   9b3c...0e72657334$
   ```

4) Use the following command to list the bloom filter files for your SSTables:

   ```
   ls -lh *Filter.db
   ```

```
ubuntu@ds201-
node1:/home/ubuntu/node/data/data/keyspace1/standard1-
9b3c...0e72657334$ ls -lh *Filter.db
-rw-rw-r-- 1 ubuntu ubuntu 305K Apr 12 13:28 mc-1-big-Filter.db
-rw-rw-r-- 1 ubuntu ubuntu  65K Apr 12 13:28 mc-2-big-Filter.db
-rw-rw-r-- 1 ubuntu ubuntu  65K Apr 12 13:29 mc-3-big-Filter.db
-rw-rw-r-- 1 ubuntu ubuntu  50K Apr 12 13:35 mc-4-big-Filter.db
```

Take note of the file sizes.

We will now decrease the probability that a bloom filter will return a false positive and see how this affects the bloom filter files sizes.

5) Execute the following command to view the current bloom filter settings:

```
DESCRIBE keyspace keyspace1;
```

```
cqlsh> DESCRIBE keyspace keyspace1;

CREATE KEYSPACE keyspace1 WITH replication = {'class':
'SimpleStrategy', 'replication_factor': '1'}  AND durable_writes
= true;

CREATE TABLE keyspace1.counter1 (
    key blob,
    column1 text,
    "C0" counter static,
    "C1" counter static,
    "C2" counter static,
    "C3" counter static,
    "C4" counter static,
    value counter,
    PRIMARY KEY (key, column1)
) WITH COMPACT STORAGE
    AND CLUSTERING ORDER BY (column1 ASC)
    AND bloom_filter_fp_chance = 0.01
    ...
    AND speculative_retry = '99PERCENTILE';

CREATE TABLE keyspace1.standard1 (
    key blob PRIMARY KEY,
    "C0" blob,
    "C1" blob,
    "C2" blob,
    "C3" blob,
```

```
    "C4" blob
) WITH COMPACT STORAGE
    AND bloom_filter_fp_chance = 0.01
    ...
    AND speculative_retry = '99PERCENTILE';
```

Note the `bloom_filter_fp_chance` is 0.01, meaning a 1% chance of a false positive. That's pretty good, but if we want an even smaller chance, we can trade space for it.

6) Execute the following command:

```
ALTER TABLE keyspace1.standard1 WITH bloom_filter_fp_chance = 0.0001;
```

Now that we have changed the `bloom_filter_fp_chance`, we must update our SSTables and associated bloom filter files.

7) Switch back to your terminal window and execute the following command:

```
/home/ubuntu/node/resources/cassandra/bin/nodetool
upgradesstables --include-all-sstables
```

`nodetool upgradesstables` rebuilds SSTables for a specified keyspace and table. We are doing this here to rebuild the Bloom Filters also. Normally this command will only upgrade sstables if the sstables are not at the most recent SSTable version; the `--include-all-sstables` flag forces the rebuild to occur. Normally you would need to run `nodetool upgradesstables` on each node. For the purposes of this exercise, we only have one node.

8) Now examine the new size of your bloom filter files.

```
ubuntu@ds201-node1:/home/ubuntu/node/data/data/keyspace1/standard1-
9b3cd4711f7111e79d8a7b0e72657334$ ls -lh *Filter.db
-rw-rw-r-- 1 ubuntu ubuntu 1.4K Apr 16 23:43 aa-16-bti-Filter.db
-rw-rw-r-- 1 ubuntu ubuntu 487K Apr 16 23:43 aa-17-bti-Filter.db
-rw-rw-r-- 1 ubuntu ubuntu 616K Apr 16 23:43 aa-18-bti-Filter.db
```

Notice the size of these files is larger. We traded space for a smaller chance of a false positive.

9) Now execute the following command:

```
ALTER TABLE keyspace1.standard1 WITH bloom_filter_fp_chance = 1.0;
```

10) And update your SSTables once again in your terminal:

```
/home/ubuntu/node/resources/cassandra/bin/nodetool upgradesstables --include-all-sstables
```

Now what is the size of your bloom filter files? Why? :)

11) Now execute the following command in your terminal:

```
/home/ubuntu/node/resources/cassandra/bin/nodetool cfstats keyspace1.standard1
```

Part of the stats include bloom filter information. Since we have not read from the `cassandra-stress` tables, the values are all zero. However, you can use these stats to tune Apache Cassandra™ if necessary.

ubuntu@ds201-node1:~$ /home/ubuntu/node/resources/cassandra/bin/nodetool cfstats keyspace1.standard1
Total number of tables: 47

```
Keyspace : keyspace1
        Read Count: 0
        Read Latency: NaN ms
        Write Count: 500000
        Write Latency: 0.036269162 ms
        Pending Flushes: 0
                Table: standard1
                SSTable count: 3
                Space used (live): 107497287
                Space used (total): 107497287
                Space used by snapshots (total): 0
                Off heap memory used (total): 0
                SSTable Compression Ratio: -1.0
                Number of partitions (estimate): 252151
                Memtable cell count: 0
                Memtable data size: 0
                Memtable off heap memory used: 0
                Memtable switch count: 14
                Local read count: 0
                Local read latency: NaN ms
                Local write count: 500000
                Local write latency: 0.028 ms
                Pending flushes: 0
                Percent repaired: 0.0
                Bytes repaired: 0.000KiB
                Bytes unrepaired: 98.525MiB
                Bytes pending repair: 0.000KiB
                Bloom filter false positives: 0
                Bloom filter false ratio: 0.00000
```

```
                Bloom filter space used: 0
                Bloom filter off heap memory used: 0
                Index summary off heap memory used: 0
                Compression metadata off heap memory used: 0
                Compacted partition minimum bytes: 180
                Compacted partition maximum bytes: 258
                Compacted partition mean bytes: 258
                Average live cells per slice (last five minutes):
NaN
                Maximum live cells per slice (last five minutes):
0
                Average tombstones per slice (last five minutes):
NaN
                Maximum tombstones per slice (last five minutes):
0
                Dropped Mutations: 0
                Failed Replication Count: null
```