DC-1 Cas1 -

```
docker run -v
/Users/neha/NehaVerma/Software/apache-cassandra-3.11.9/conf/cassandra.yaml:/etc/cas
sandra/cassandra.yaml --name cas1 -p 9042:9042 -e CASSANDRA_CLUSTER_NAME=MyCluster
-e CASSANDRA_ENDPOINT_SNITCH=GossipingPropertyFileSnitch -e
CASSANDRA_DC=datacenter1 -d cassandra
```

DC-1 Cas1 - With 3000

```
docker run -v
/Users/neha/NehaVerma/Software/cassandra.yaml:/etc/cassandra/cassandra.yaml --name
cas1 -p 9042:9042 -e CASSANDRA_CLUSTER_NAME=MyCluster -e
CASSANDRA_ENDPOINT_SNITCH=GossipingPropertyFileSnitch -e CASSANDRA_DC=datacenter1
-d cassandra
```

IpAddress -

```
docker inspect --format='{{ .NetworkSettings.IPAddress }}'
cas1
```

Result

172.17.0.2

DC-1 Cas2 - with 10

```
docker run -v
/Users/neha/NehaVerma/Software/apache-cassandra-3.11.9/conf/cassandra.yaml:/etc/cas
sandra/cassandra.yaml --name cas2 -e CASSANDRA_SEEDS="172.17.0.2" -e
CASSANDRA_CLUSTER_NAME=MyCluster -e
CASSANDRA_ENDPOINT_SNITCH=GossipingPropertyFileSnitch -e CASSANDRA_DC=datacenter1
-d cassandra
```

DC-1 Cas3 - with 10

```
docker run -v
/Users/neha/NehaVerma/Software/apache-cassandra-3.11.9/conf/cassandra.yaml:/etc/cas
sandra/cassandra.yaml --name cas3 -e CASSANDRA_SEEDS="172.17.0.2" -e
CASSANDRA_CLUSTER_NAME=MyCluster -e
CASSANDRA_ENDPOINT_SNITCH=GossipingPropertyFileSnitch -e CASSANDRA_DC=datacenter1
-d cassandra
```

DC-2 Cas-4

```
docker run -v
/Users/neha/NehaVerma/Software/apache-cassandra-3.11.9/conf/cassandra.yaml:/etc/cas
sandra/cassandra.yaml --name cas4 -e CASSANDRA_SEEDS="172.17.0.2" -e
```

```
CASSANDRA_CLUSTER_NAME=MyCluster -e
CASSANDRA_ENDPOINT_SNITCH=GossipingPropertyFileSnitch -e CASSANDRA_DC=datacenter2
-d cassandra
```

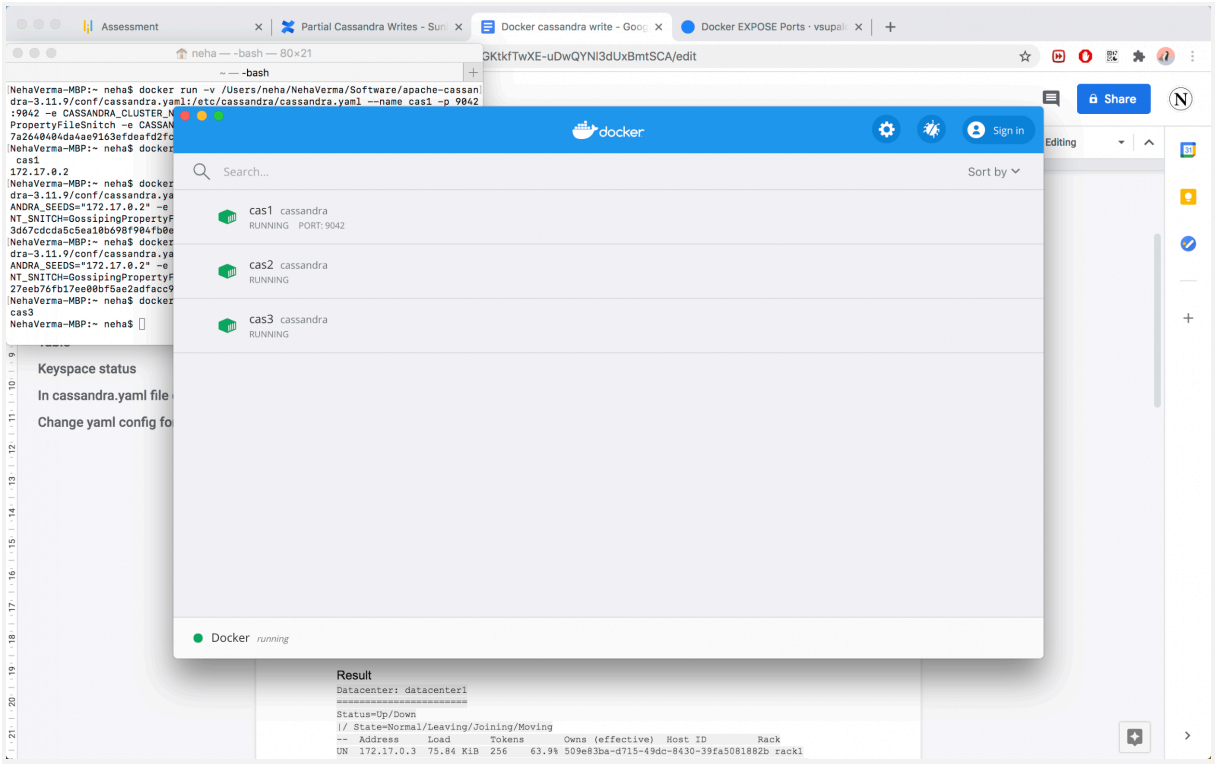
Status

```
docker exec -ti cas1 nodetool status
```

Result

```
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address      Load        Tokens       Owns (effective)  Host ID                               Rack
UN  172.17.0.3   75.84 KiB   256     63.9% 509e83ba-d715-49dc-8430-39fa5081882b  rack1
UN  172.17.0.2   127.55 KiB  256   67.7%  9faca744-4593-425d-99a0-d3c3602e4e9d   rack1

Datacenter: datacenter2
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address      Load        Tokens       Owns (effective)  Host ID    Rack
UN  172.17.0.4   90 KiB      256     68.4%     f14f2365-42be-4c59-867d-d317b485a48d  rack1
```



Connect cqlsh

```
docker exec -ti cas1 cqlsh
docker exec -ti cas2 cqlsh
docker exec -ti cas3 cqlsh
```

Keyspace

```
CREATE KEYSPACE test WITH replication = {'class' :
'NetworkTopologyStrategy', 'datacenter1' : 1, 'datacenter2' : 1};
use test;
```

Follow - `CREATE KEYSPACE test WITH replication =`
`{'class':'SimpleStrategy', 'replication_factor':2};`
`use test;`

Table

```
CREATE TABLE user(name text, id text, PRIMARY KEY (name));

Select * from user;

truncate user;

SELECT COUNT(*) FROM user;

Select * from user where name='user-1';
```

Keyspace status

```
docker exec -ti cas1 nodetool status test
```

docker stop cas1 cas2 cas3

docker start cas1 cas2 cas3

In cassandra.yaml file change timeout (ms)

```
write_request_timeout_in_ms: 10

Default -

write_request_timeout_in_ms: 3000
```

Change yaml config for image

```
docker container run -v
/Users/neha/NehaVerma/Software/apache-cassandra-3.11.9/conf/cassandra.yaml:/etc/cassan
dra/cassandra.yaml your_cassandra_image_name
```

Run Java Application

Scenario 1:

```
ConsistencyLevel quorum
BatchStatement.Type.UNLOGGED
```

Error :

1. WriteFailureException: Cassandra failure during write query at consistency QUORUM (2 responses were required but only 1 replica responded, 1 failed)

Observation:

DC-1 cas1,cas2 : Data Got inserted + updated

DC-2 cas3 : Data Got inserted + updated

2. cas2 - down

   NoHostAvailableException: All host(s) tried for query failed (tried: localhost/0:0:0:0:0:0:0:1:9042(com.datastax.driver.core.exceptions.UnavailableException: Not enough replicas available for query at consistency QUORUM (2 required but only 1 alive)))

Observation:

DC-1 cas1,cas2 : Data not inserted + updated

DC-2 cas3 : Data not inserted + updated

Scenario 2:

```
ConsistencyLevel quorum
BatchStatement.Type.LOGGED
```

```
getWriteType : BATCH
Give success
```

Error:

1. WriteTimeoutException: Cassandra timeout during write query at consistency ONE (1 replica were required but only 0 acknowledged the write)

Observation:

DC-1 cas1,cas2 : Data Got inserted + updated

DC-2 cas3 : Data Got inserted + updated

2. Cassandra timeout during write query at consistency QUORUM (2 replica were required but only 1 acknowledged the write)

Observation :

getWriteType : BATCH

DC-1 cas1 : 3000 : Data Got inserted + updated

DC-1 cas2 cas3: 10 : Data Got inserted + updated

Content State Update Api

CREATE KEYSPACE sunbird_courses WITH replication = {'class': 'SimpleStrategy', 'replication_factor': '2'} AND durable_writes = true;

```
CREATE TABLE sunbird_courses.course_batch (
    courseid text,
    batchid text,
    cert_templates map<text, frozen<map<text, text>>>,
    createdby text,
    createddate text,
    createdfor list<text>,
    description text,
    enddate text,
    enrollmentenddate text,
    enrollmenttype text,
    mentors list<text>,
    name text,
    startdate text,
    status int,
    tandc boolean,
    updateddate text,
    PRIMARY KEY (courseid, batchid)
) WITH CLUSTERING ORDER BY (batchid ASC)
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold':
'32', 'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '64', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND dclocal_read_repair_chance = 0.1
    AND default_time_to_live = 0
```

```
   AND gc_grace_seconds = 864000
   AND max_index_interval = 2048
   AND memtable_flush_period_in_ms = 0
   AND min_index_interval = 128
   AND read_repair_chance = 0.0
   AND speculative_retry = '99PERCENTILE';


CREATE TABLE sunbird_courses.user_enrolments (
    userid text,
    courseid text,
    batchid text,
    active boolean,
    addedby text,
    certificates list<frozen<map<text, text>>>,
    certstatus int,
    completedon timestamp,
    completionpercentage int,
    contentstatus map<text, int>,
    datetime timestamp,
    enrolleddate text,
    issued_certificates list<frozen<map<text, text>>>,
    lastreadcontentid text,
    lastreadcontentstatus int,
    progress int,
    status int,
    PRIMARY KEY (userid, courseid, batchid)
) WITH CLUSTERING ORDER BY (courseid ASC, batchid ASC)
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class':
'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32',
'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '64', 'class':
'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND dclocal_read_repair_chance = 0.1
    AND default_time_to_live = 0
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair_chance = 0.0
    AND speculative_retry = '99PERCENTILE';


CREATE TYPE sunbird_courses.question (
    id text,
    assess_ts timestamp,
    max_score double,
    score double,
    type text,
    title text,
    resvalues list<frozen<map<text, text>>>,
    params list<frozen<map<text, text>>>,
    description text,
    duration decimal
);
```

```
CREATE TABLE sunbird_courses.user_content_consumption (
    userid text,
    courseid text,
    batchid text,
    contentid text,
    completedcount int,
    datetime timestamp,
    lastaccesstime text,
    lastcompletedtime text,
    lastupdatedtime text,
    progress int,
    status int,
    viewcount int,
    PRIMARY KEY (userid, courseid, batchid, contentid)
) WITH CLUSTERING ORDER BY (courseid ASC, batchid ASC, contentid ASC)
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class':
'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32',
'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '64', 'class':
'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND dclocal_read_repair_chance = 0.1
    AND default_time_to_live = 0
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair_chance = 0.0
    AND speculative_retry = '99PERCENTILE';


CREATE TABLE sunbird_courses.assessment_aggregator (
   course_id text,
   batch_id text,
   user_id text,
   content_id text,
   attempt_id text,
   created_on timestamp,
   grand_total text,
   last_attempted_on timestamp,
   question list<frozen<question>>,
   total_max_score double,
   total_score double,
   updated_on timestamp,
   PRIMARY KEY (course_id, batch_id, user_id, content_id, attempt_id)
) WITH CLUSTERING ORDER BY (batch_id ASC, user_id ASC, content_id ASC, attempt_id ASC)
   AND bloom_filter_fp_chance = 0.01
   AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
   AND comment = ''
   AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold':
'32', 'min_threshold': '4'}
   AND compression = {'chunk_length_in_kb': '64', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
   AND crc_check_chance = 1.0
   AND dclocal_read_repair_chance = 0.1
   AND default_time_to_live = 0
   AND gc_grace_seconds = 864000
   AND max_index_interval = 2048
   AND memtable_flush_period_in_ms = 0
   AND min_index_interval = 128
```

```
  AND read_repair_chance = 0.0
  AND speculative_retry = '99PERCENTILE';
```

```
INSERT INTO sunbird_courses.course_batch(courseid, batchid) VALUES
('do_113168060882804736111 3','0131680712715878401');
```

Select * from `sunbird_courses.course_batch;`

```
Select count(*) from sunbird_courses.user_content_consumption;

truncate sunbird_courses.user_content_consumption;
```

```
-----------------------------------code------------------------------
long startTime = System.currentTimeMillis();
logger.info(requestContext,
       "Cassandra Service batchInsertLogged method started at ==" +
startTime);

Session session = connectionManager.getSession(keyspaceName);
Response response = new Response();
BatchStatement batchStatement = new
BatchStatement(BatchStatement.Type.LOGGED);
batchStatement.setConsistencyLevel(ConsistencyLevel.QUORUM);
ResultSet resultSet = null;

try {
 PreparedStatement preparedStatement = session.prepare("insert into
sunbird_courses.user_content_consumption(courseid, batchid, userid,
contentid) values (?, ?, ?, ?)");
 int i = 1;
 while (i <= 10000) {

batchStatement.add(preparedStatement.bind("do_113168060882804736111 3",
"0131680712715878401", "874ed8a5-782e-4f6c-8f36-e0288455901e",
"do_1125984304256122881 6-" + i));
   i++;
 }
 resultSet = session.execute(batchStatement);
 response.put(Constants.RESPONSE, Constants.SUCCESS);
} catch (QueryExecutionException
       | QueryValidationException
       | NoHostAvailableException
       | IllegalStateException e) {
 logger.info(requestContext, "Cassandra Batch Insert Failed." +
e.getMessage(), e);
 if (e.getClass() == WriteTimeoutException.class &&
((WriteTimeoutException) e).getWriteType().name().equals("BATCH") ||
((WriteTimeoutException) e).getWriteType().name().equals("SIMPLE"))
```

```
    response.put(Constants.RESPONSE, Constants.SUCCESS);
  else {
    throw new ProjectCommonException(
            ResponseCode.SERVER_ERROR.getErrorCode(),
            ResponseCode.SERVER_ERROR.getErrorMessage(),
            ResponseCode.SERVER_ERROR.getResponseCode());
  }
}
logQueryElapseTime("batchInsertLogged", startTime);
return response;
```

JavaScript

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript While Loop</h2>

<p id="demo"></p>

<script>
var text = "";
var i = 0;
while (i < 1000) {
  text += "<br><br>" +"{"+ '"contentId"'  +":" + "\"do_11259843042561228816-"+ i +
"\"," +'"batchId"' +":" + '"0131680712715878401"'+ "," +'"status"'+":" + "2" + ","
+  '"courseId"'+":" + '"do_113168060888280473611113"' + "},";
  i++;
}
document.getElementById("demo").innerHTML = text;
</script>

</body>
</html>
```

Delete Query

delete from user_content_consumption where userid='95e4942d-cbe8-477d-aebd-ad8e6de4bfc8' and courseid='do_113168060888280473611113' and batchid='0131680712715878401'; (edited)

2:52

select * from user_content_consumption limit 2;