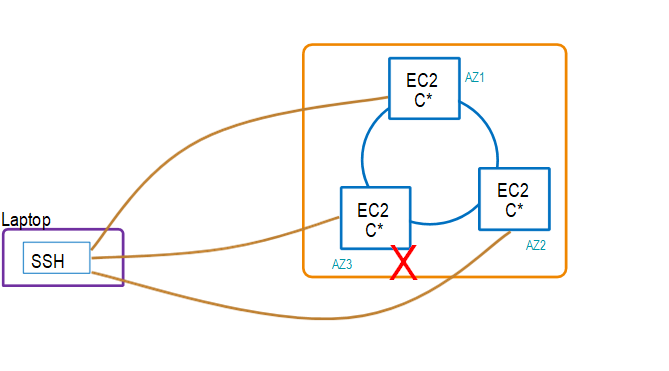
**Lab Brief**

Course: Big Data Management on Cloud

**Cassandra Setup | Master-less arch concepts**

(Install multi-node Cassandra cluster, induce failure, create a keyspace/table and access from the client)

Diagram:

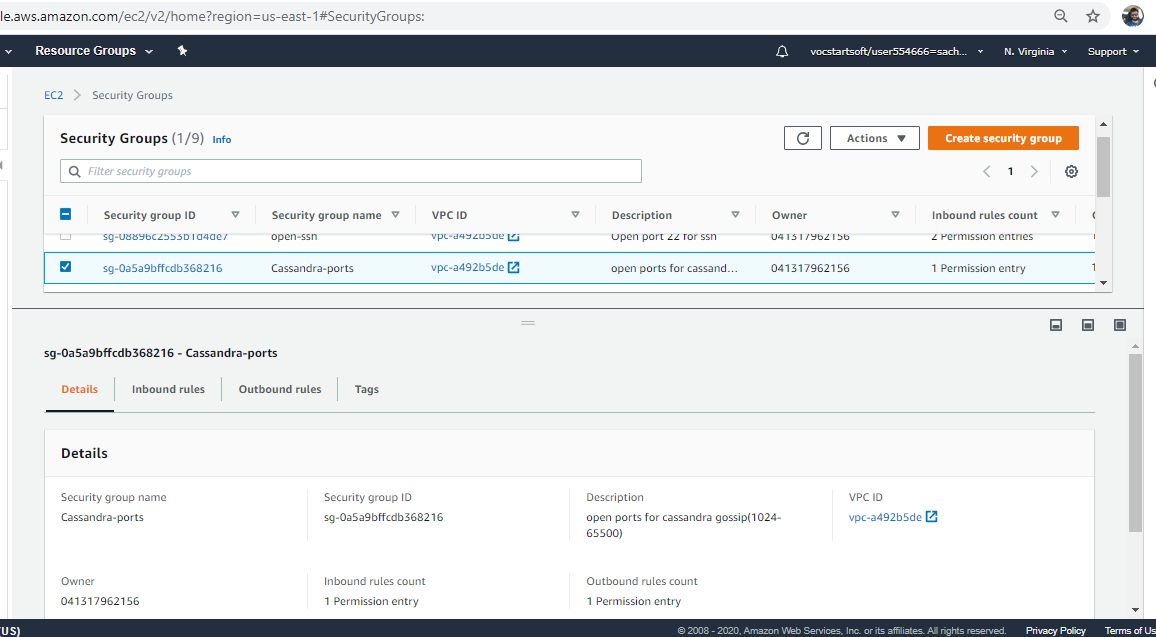


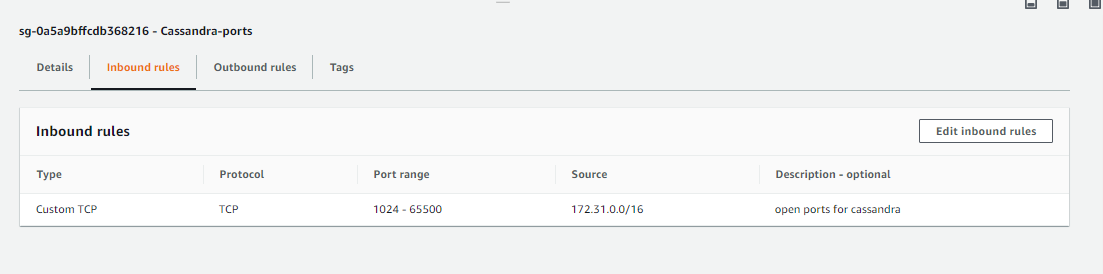
Steps to be followed :-

1. Create a new security group by the name "cassandra-ports"

"Custom TCP"and port range of "1024-65500"

"VPC CIDR block" as the allowable traffic source





2. Create 3 EC2 instances (t2-small) using the 7 step workflow

a) Use the "SKL-Ubuntu-Cassandra" (community AMI) in AZ1, AZ2, AZ3

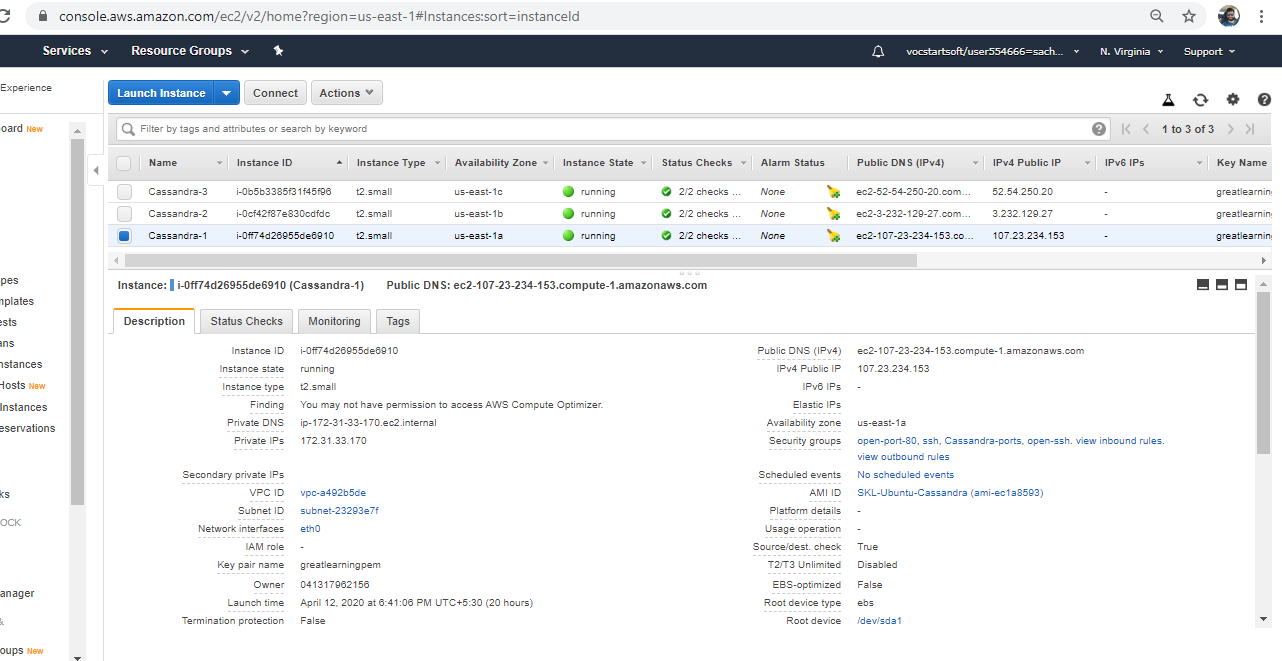
b) The AMI contains - Ubuntu 16.04, JDK8, Python2.7 & unconfigured Cassandra

c) Assign the "cassandra-ports" SG

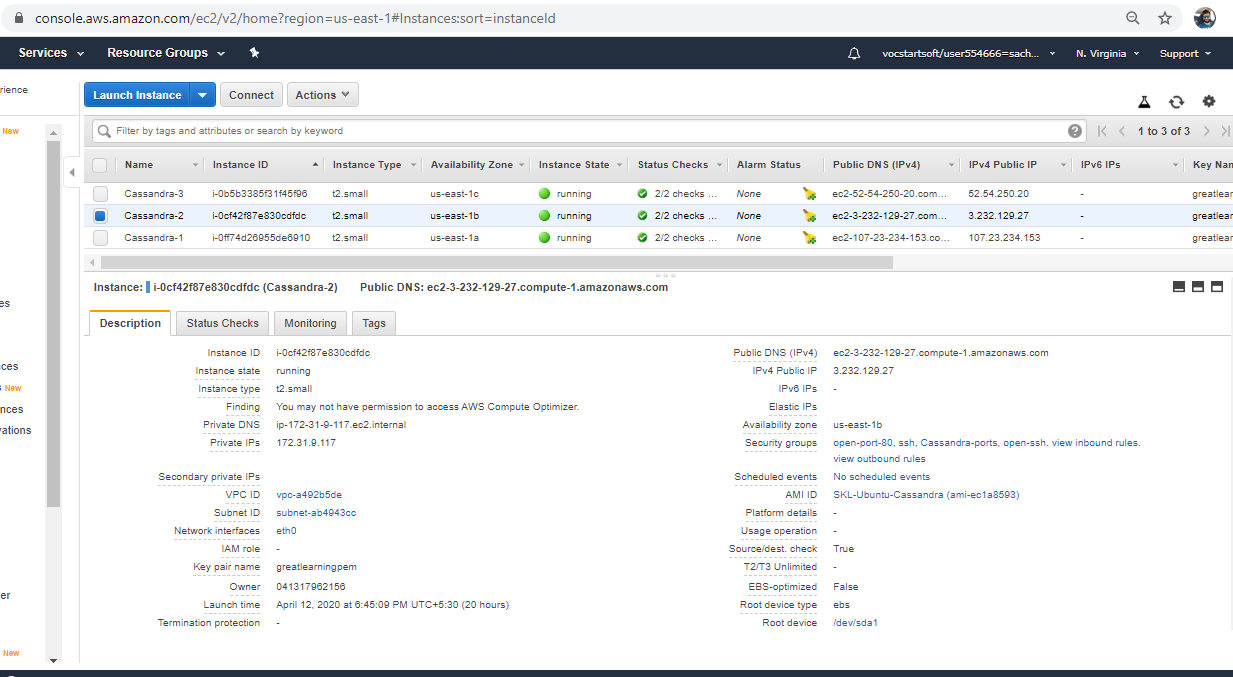
d) Download a new PEM file

e) Open 3 terminal windows, one for each instance

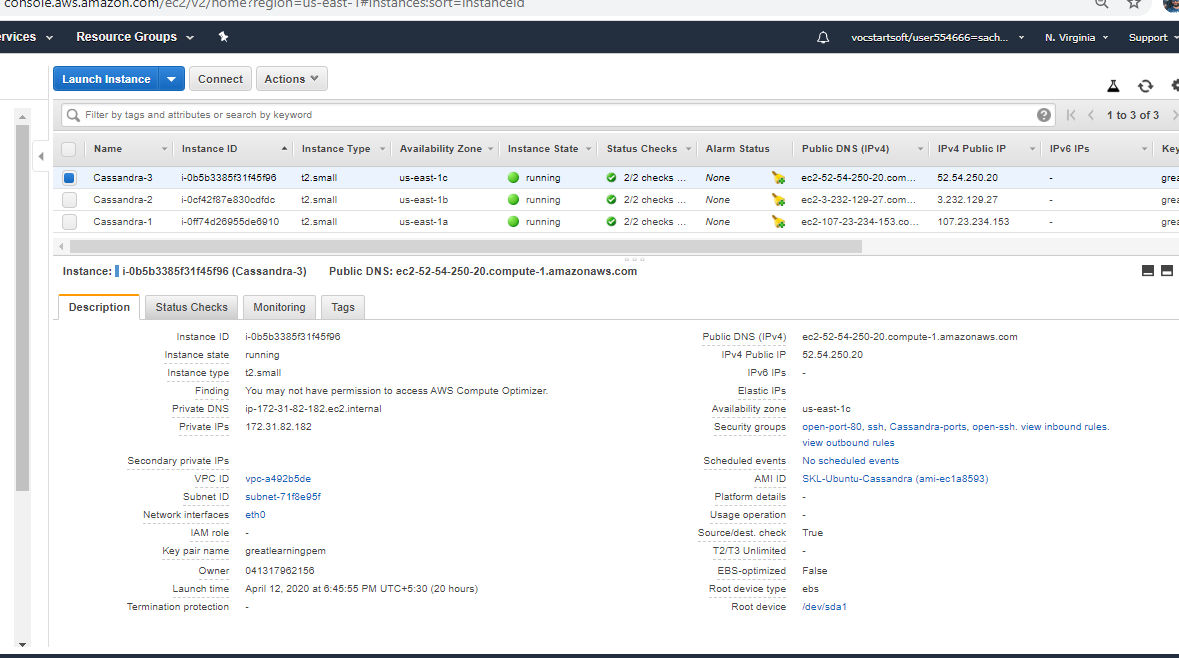
Cassandra-1



Cassandra-2



Cassandra-3



3. Follow the steps to configure Cassandra (repeat this in all AZ)

a) Go to the "conf" folder and update the cassandra.yaml (reference 1)

b) Start the cassandra daemon (reference 2)

c) Issue the nodetool command to ensure that the startup good

**For Cassandra-1**

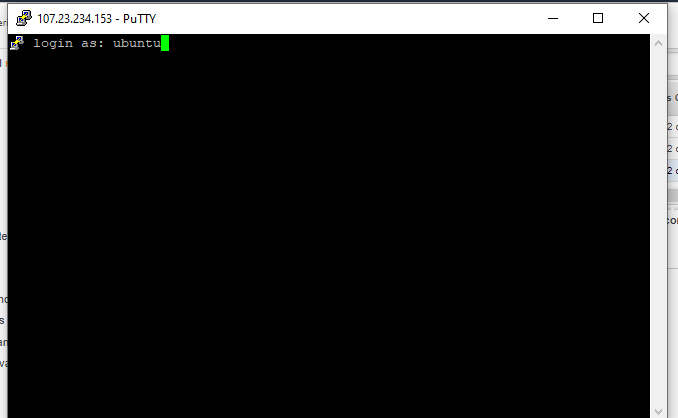
sed -i 's=MOD\_CLUSTER\_NAME=GL-Cluster=g' cassandra.yaml

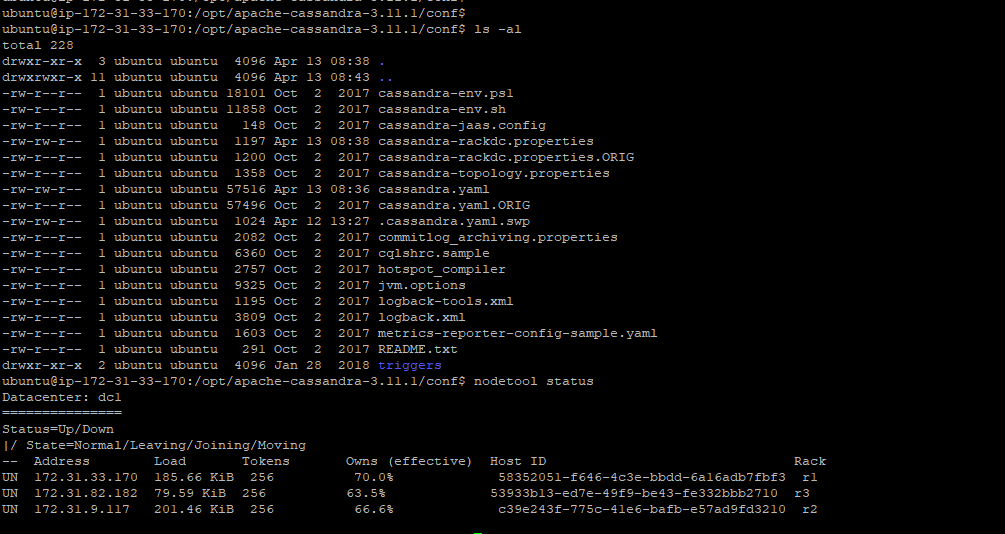
sed -i 's=MOD\_IP\_ADDRESS=172.31.33.170=g' cassandra.yaml

sed -i 's=MOD\_SEED\_LIST=172.31.33.170=g' cassandra.yaml

sed -i 's=MOD\_DATACENTER=dc1=g' cassandra-rackdc.properties

sed -i 's=MOD\_RACK=r1=g' cassandra-rackdc.properties





**For Cassandra-2**

sed -i 's=MOD\_CLUSTER\_NAME=GL-Cluster=g' cassandra.yaml

sed -i 's=MOD\_IP\_ADDRESS= 172.31.9.117=g' cassandra.yaml

sed -i 's=MOD\_SEED\_LIST=172.31.33.170=g' cassandra.yaml

sed -i 's=MOD\_DATACENTER=dc1=g' cassandra-rackdc.properties

sed -i 's=MOD\_RACK=r2=g' cassandra-rackdc.properties

**For Cassandra-3**

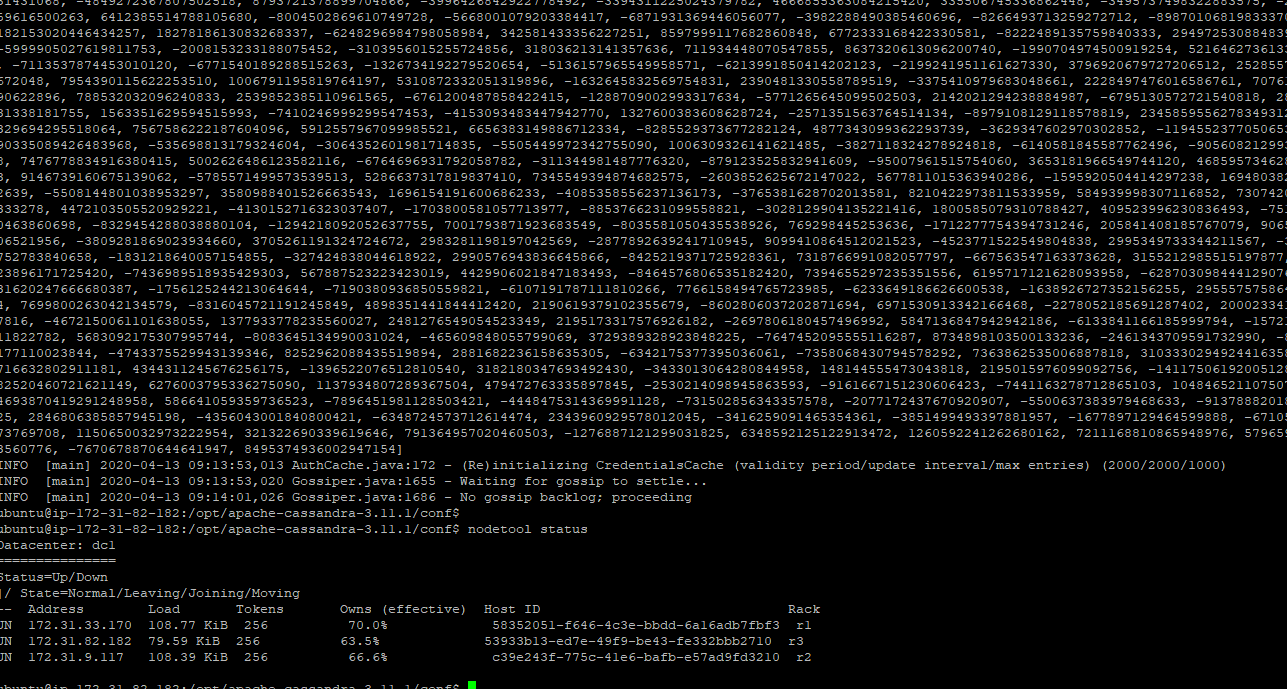
sed -i 's=MOD\_CLUSTER\_NAME=GL-Cluster=g' cassandra.yaml

sed -i 's=MOD\_IP\_ADDRESS= 172.31.82.182=g' cassandra.yaml

sed -i 's=MOD\_SEED\_LIST=172.31.33.170=g' cassandra.yaml

sed -i 's=MOD\_DATACENTER=dc1=g' cassandra-rackdc.properties

sed -i 's=MOD\_RACK=r3=g' cassandra-rackdc.properties



How to do it? – 2

ALTER KEYSPACE "system\_auth" WITH REPLICATION = {'class':'NetworkTopologyStrategy', 'dc1':3};

CREATE KEYSPACE IF NOT EXISTS starfleet WITH replication = {'class':'NetworkTopologyStrategy', 'dc1':3};

CREATE TABLE starfleet.user (

user\_id VARCHAR,

location VARCHAR,

display\_name VARCHAR,

first\_name VARCHAR,

last\_name VARCHAR,

PRIMARY KEY (user\_id, location)

);

Record 1

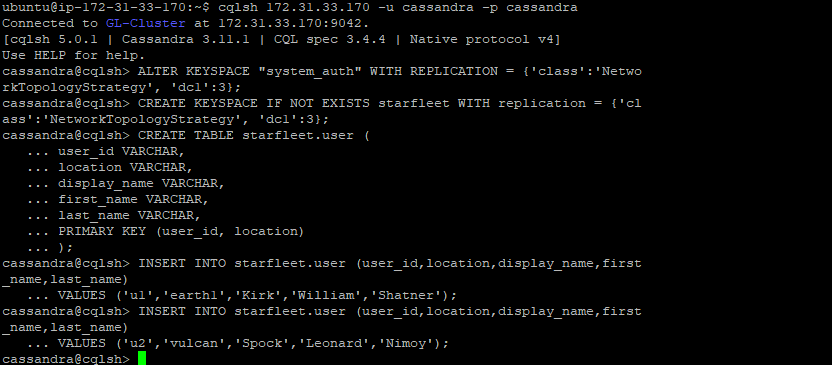
INSERT INTO starfleet.user (user\_id,location,display\_name,first\_name,last\_name)

VALUES ('u1','earth1','Kirk','William','Shatner');

Record 2

INSERT INTO starfleet.user (user\_id,location,display\_name,first\_name,last\_name)

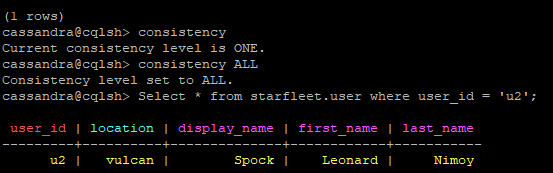
VALUES ('u2','vulcan','Spock','Leonard','Nimoy');

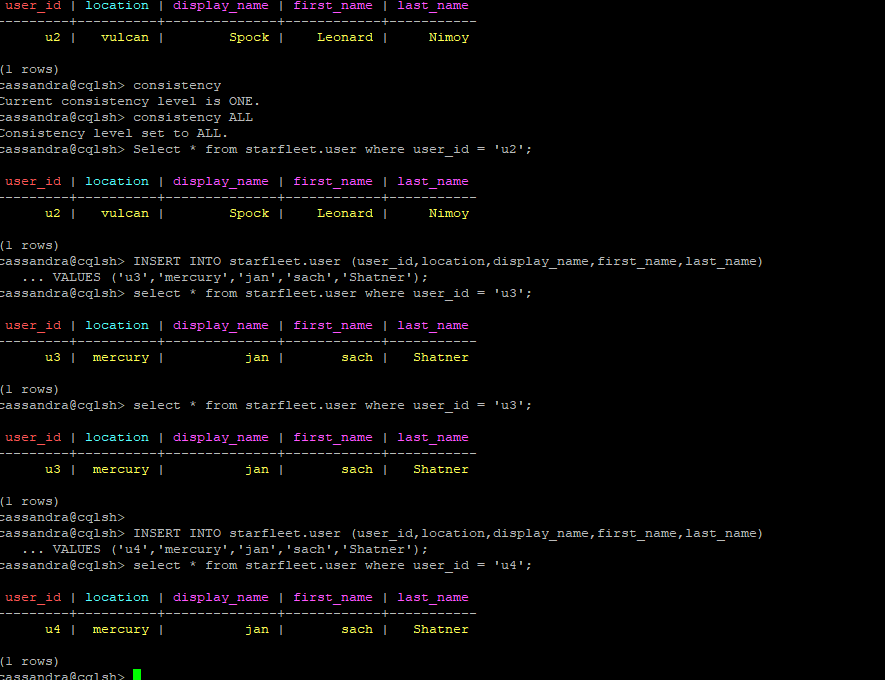


Check the consistency level (CL) - CONSISTENCY

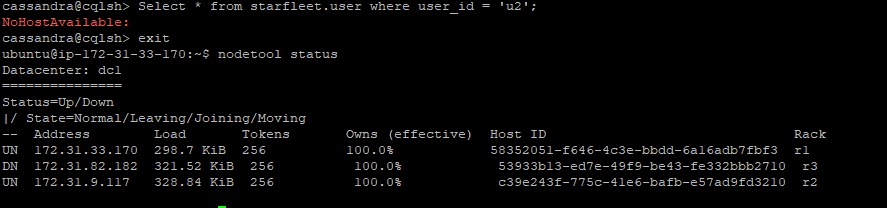
Change the CL=ALL - CONSISTENCY ALL

Select \* from starfleet.user where user\_id = 'u2';





Stop the Cassandra-3 daemon



Now run select query with consistency -**ALL**

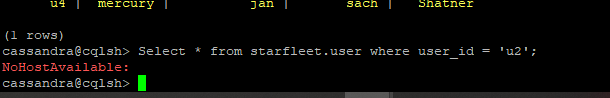
**Result** – NoHostAvailable

**Consistency** =ALL

**RF**=3

%**tolerance** = 0

Cassandra can’t tolerate any node down



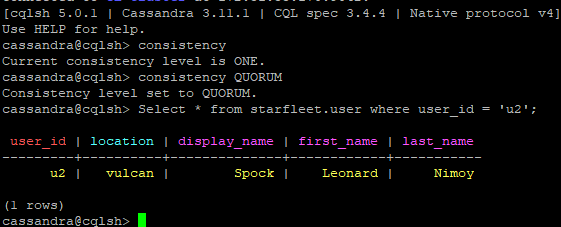
**Change consistency to QUORUM**

We are getting result because

**RF**=3

**Quorum** = 3/2 +1 = 2

**%Tolerance** = 33%, we can tolerate 1 node down



How to do it? - 3 (Advanced)

**Q1**: Which of the following will give a wide row? Justify your response.

PRIMARY KEY((event\_type, date), created\_hh, created\_min, created\_nn)

OR

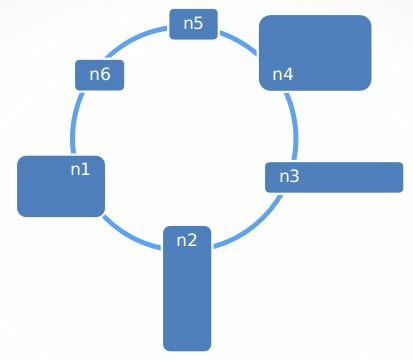
PRIMARY KEY((event\_type), date, created\_hh, created\_min, created\_nn)

**Answer**-

PRIMARY KEY((event\_type), date, created\_hh, created\_min, created\_nn)

Will give wide row because,

All events for a same type resides in a single partition, this can lead to wide rows. Suppose there are 7000 records for event type ”FIRE”, all 7000 records with different dates will be stored in the single partition results in a row with over 7000 combinations.

**Q2**: Look at the picture below and argue for or against such a topology. Assume y-axis is memory and x-axis is

cpu. So n3 has more cpu than memory.

**Answer**-

PRIMARY KEY