**Assignment No 10**

**PRN No : 2020BTECS00092**

**Batch : T3**

**Title:** Setup a multi-node Cassandra Cluster (Take 3 machines in lab or

on single machine). Give your PRN as cluster name. Follow the

steps given in below link

**Procedure:**

Requirements :

1. We start by installing the Cassandra 3.11 software
2. We would also need Java8 and Python2.7
3. We need to add the locations to the PATH and as JAVA\_HOME

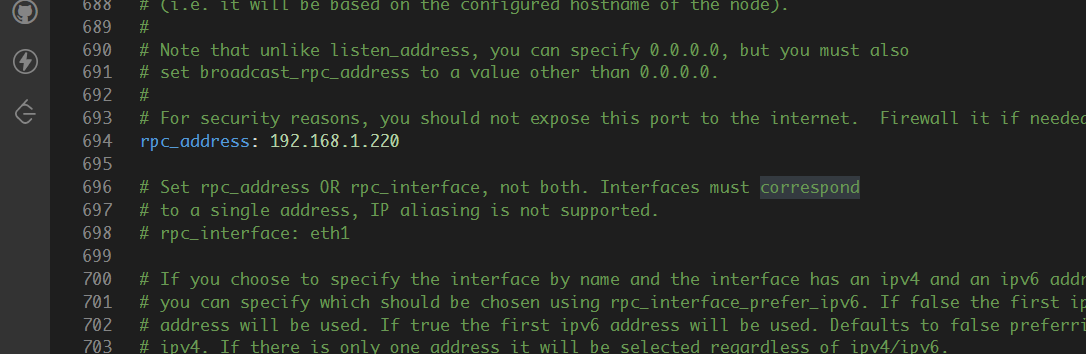
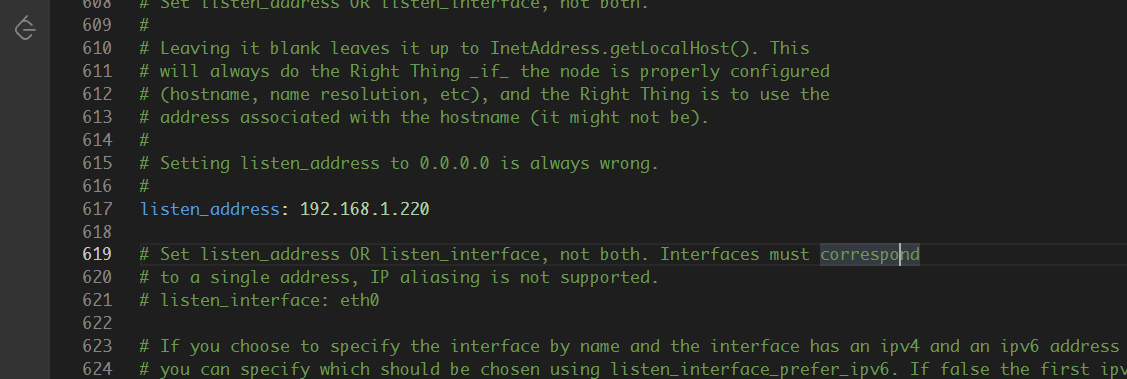
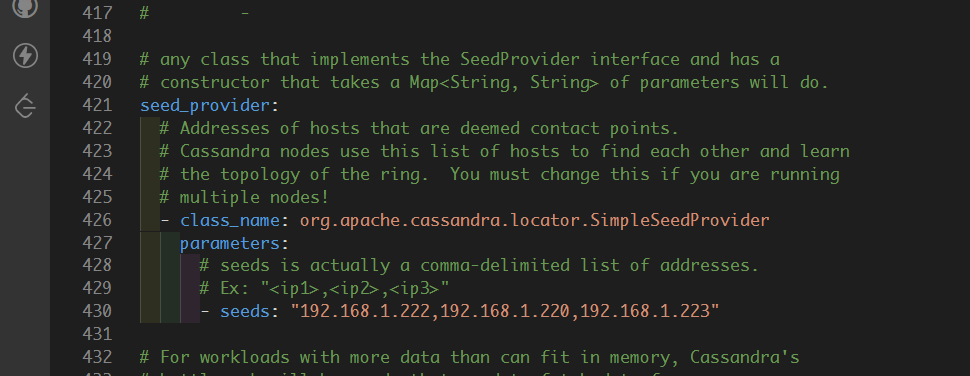
* <https://extendit.us/steps-configure-multiple-nodes-cassandra-single-windows-machine>

step 1 :

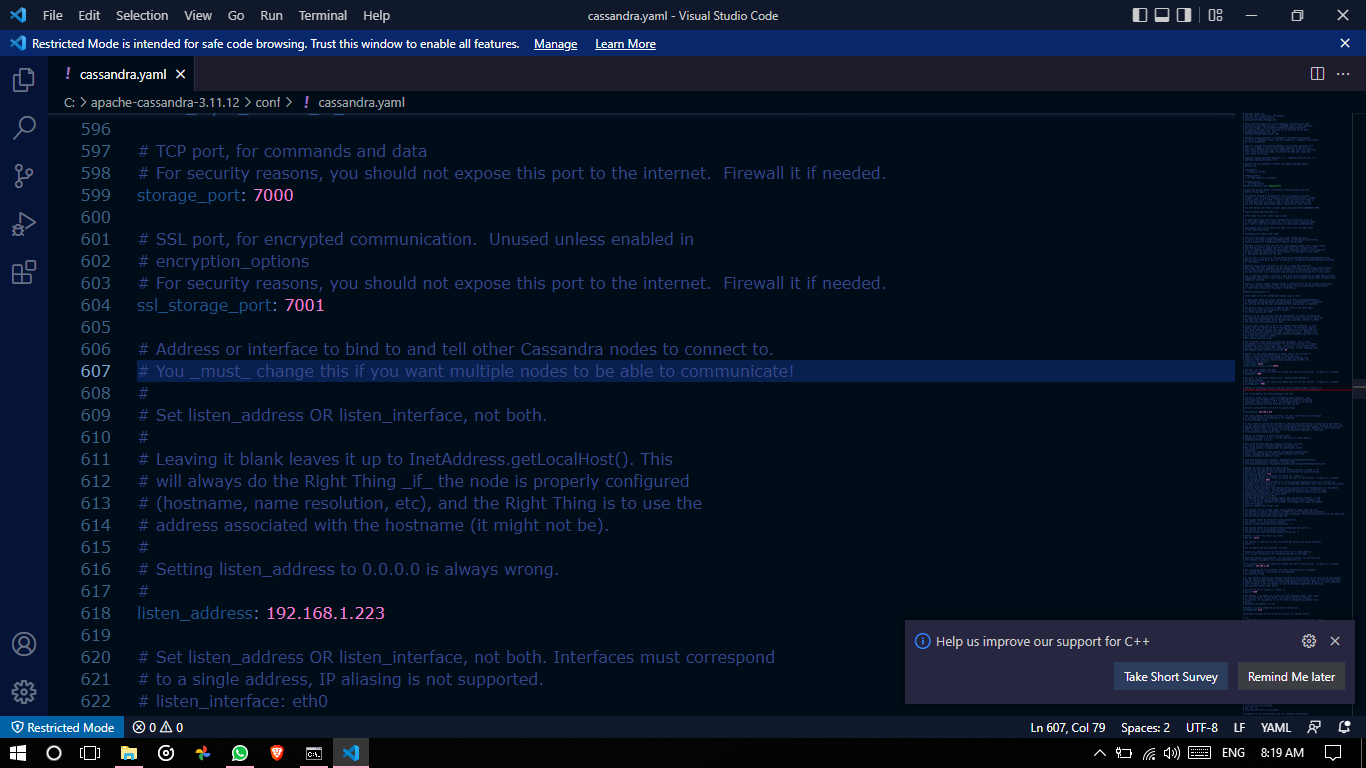
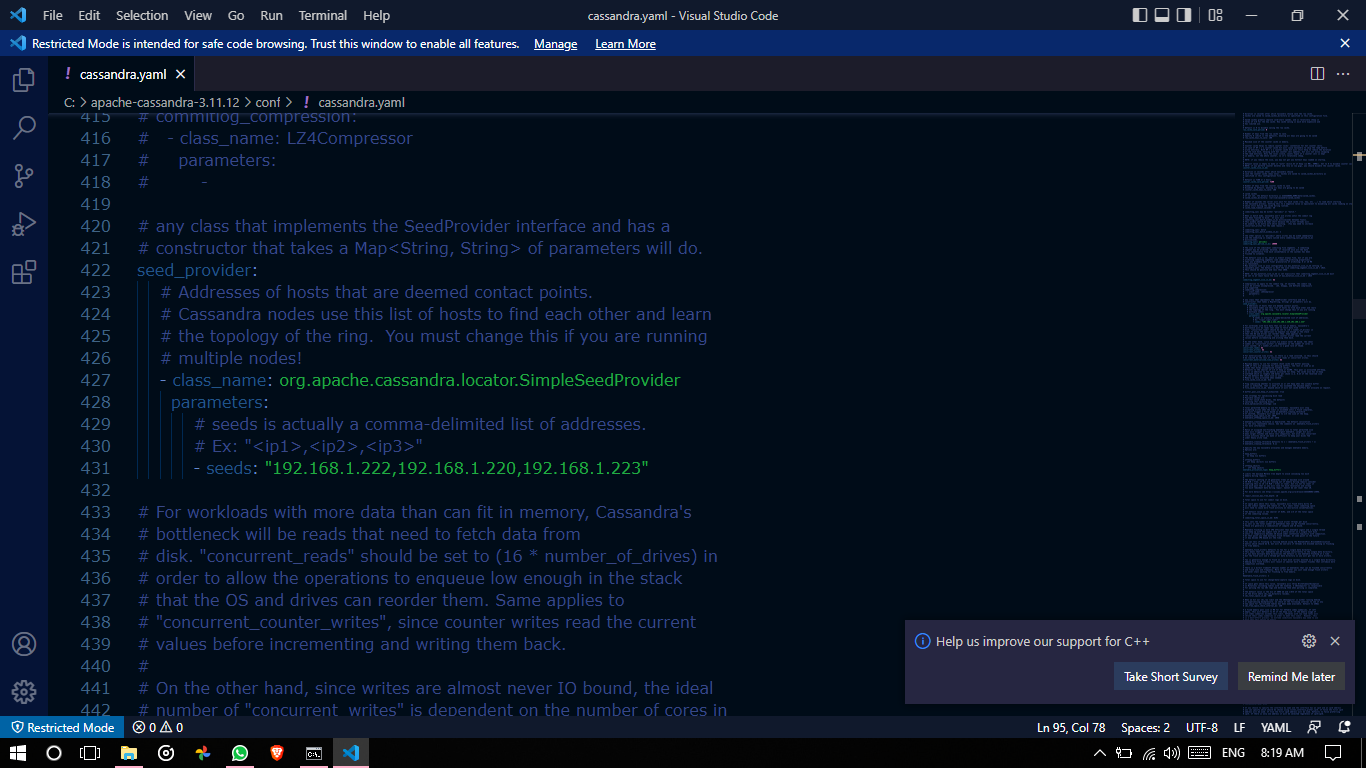
SETUP AT NODE 1:

We change the conf/cassandra.yaml

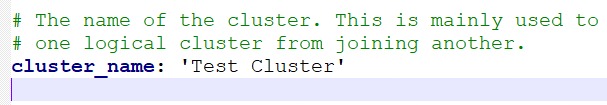
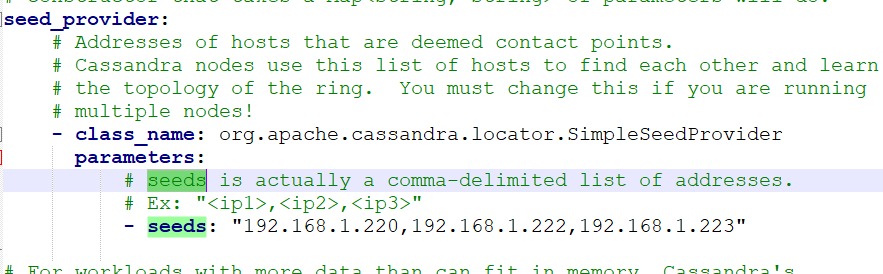
We edit these 3 values on all 3 nodes : listen\_address , rpc\_address , seeds



In Node 2:



In Node 3 :



We can now start our cluster: Going to location: C:\Cassandra\apache-cassandra-3.11.12\bin

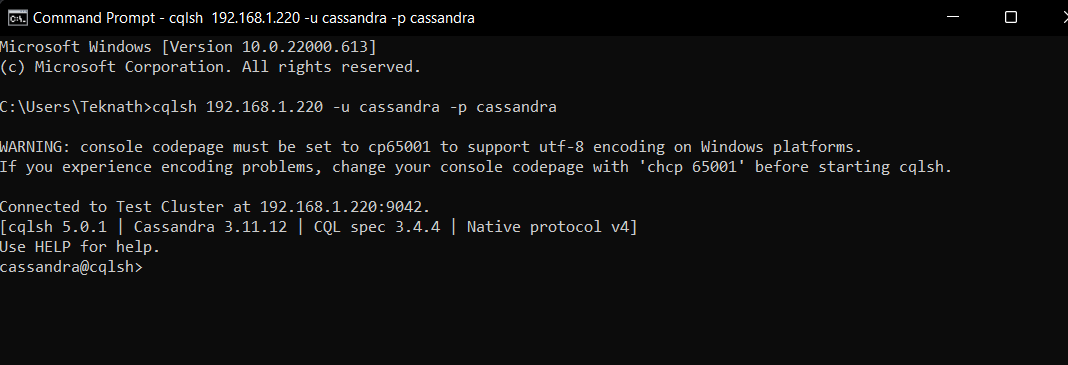
CMD : We run: cassandra.bat -f

If Java8 and Python 2.7 is installed, the cluster starts up.

Now we demonstrate by using

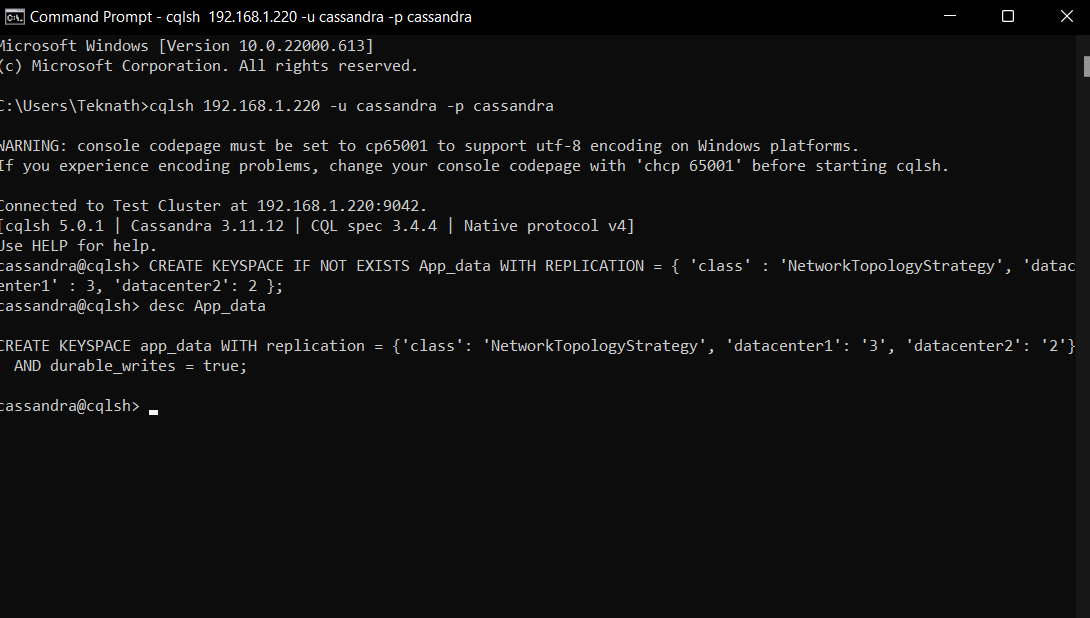
cqlsh 192.168.1.220 -u cassandra -p Cassandra

These are the default credentials over Node 1.

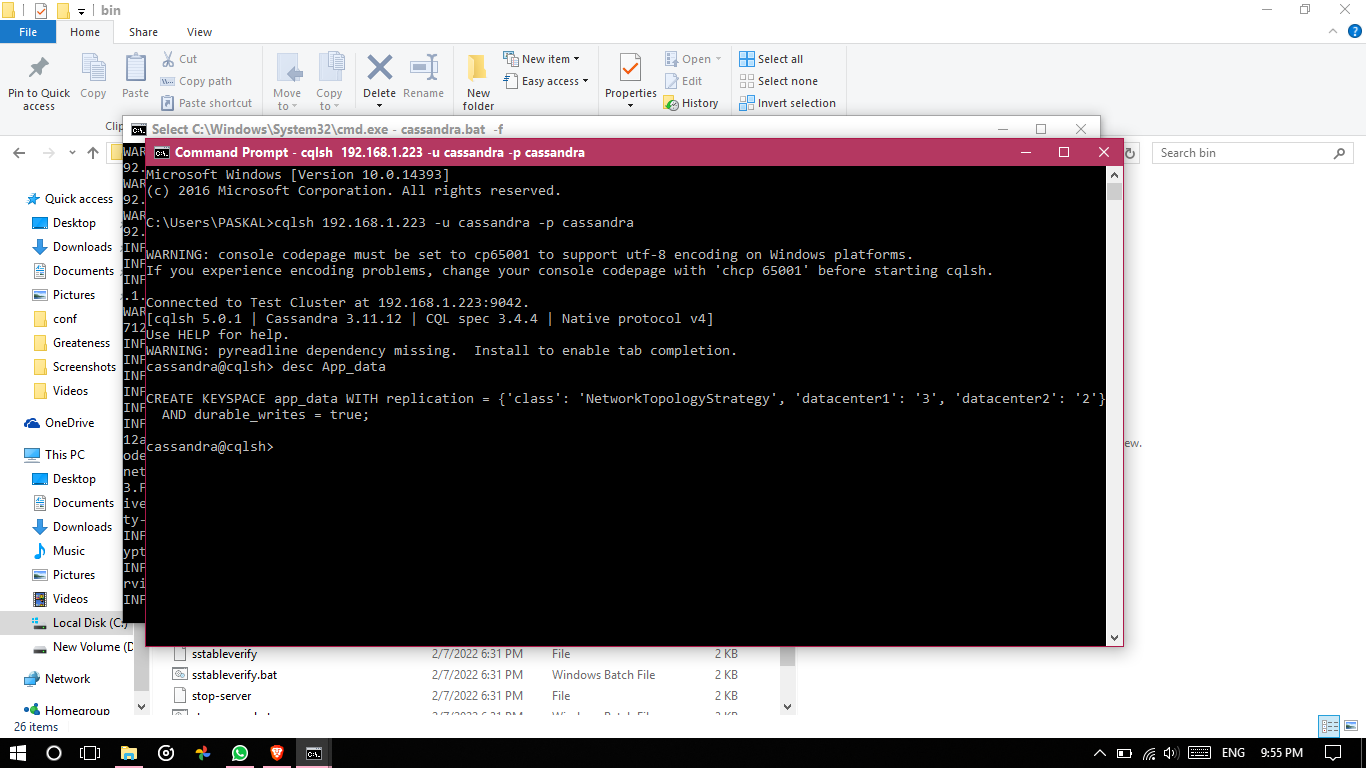


cassandra@cqlsh> CREATE KEYSPACE IF NOT EXISTS App\_data WITH REPLICATION = { 'class' : 'NetworkTopologyStrategy', 'datacenter1' : 3, 'datacenter2': 2 };

At Node 1:



At node 2:



So our schema is replicated at all node 1,2,3.

Now, we build a Temperature-Webapp On a click, we get the current temperature and store it onto Cassandra. Further, we also show all the possible temperatures stored in it.

We would be interacting with Cassandra Cluster through Express Js and make APIs to interact

**Table schema:**

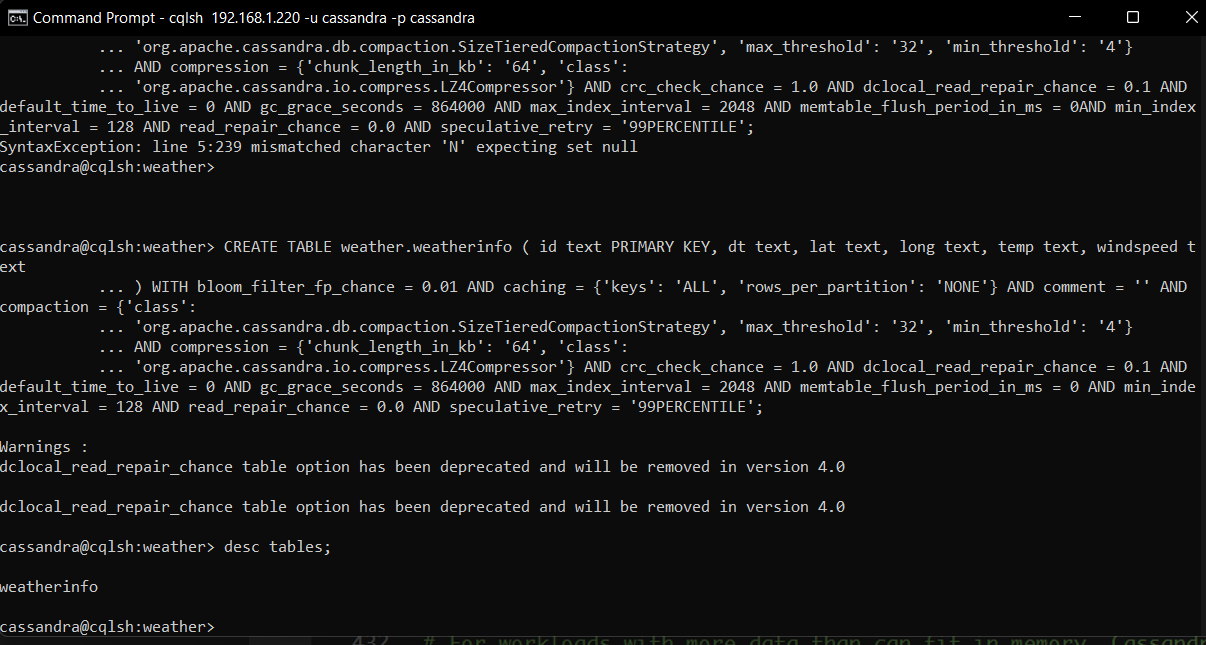
CREATE TABLE weather.weatherinfo ( id text PRIMARY KEY, dt text, lat text, long text, temp text, windspeed text

) WITH 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';



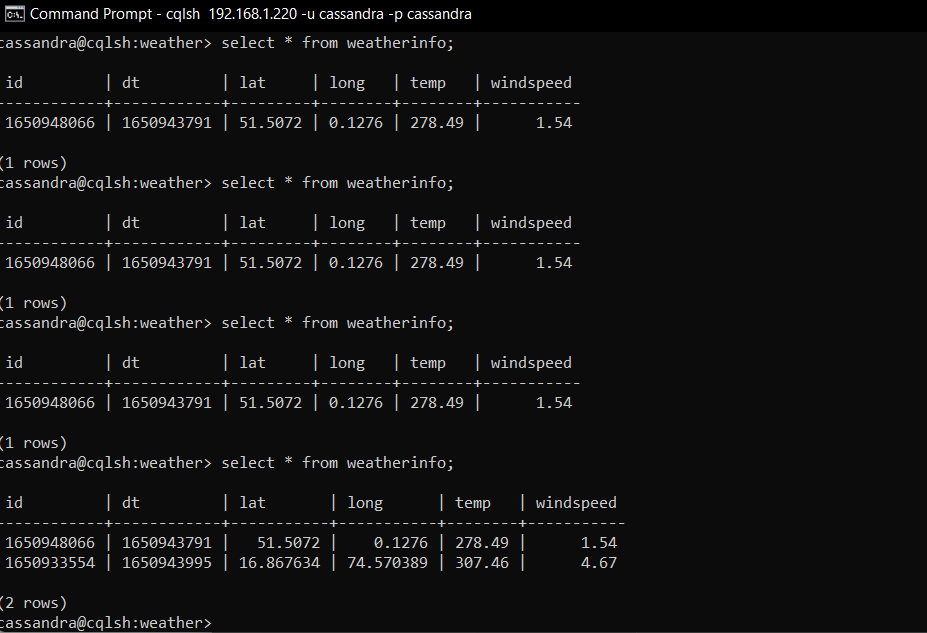
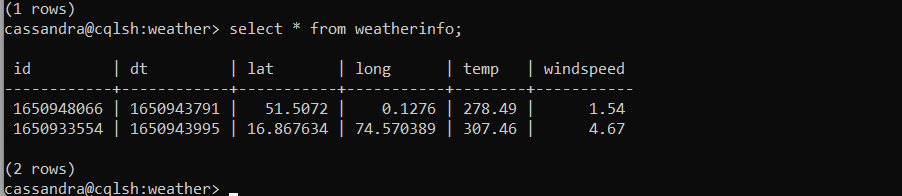
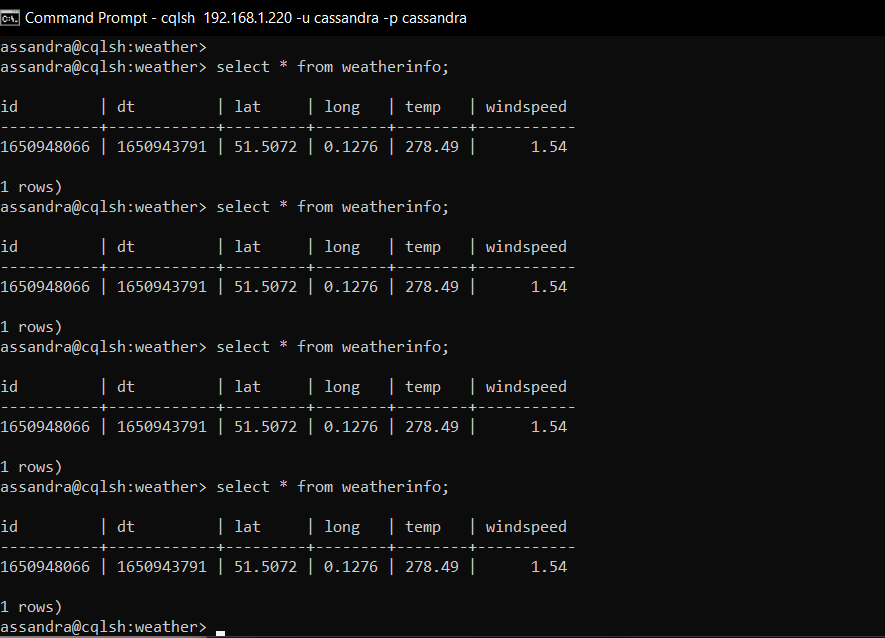
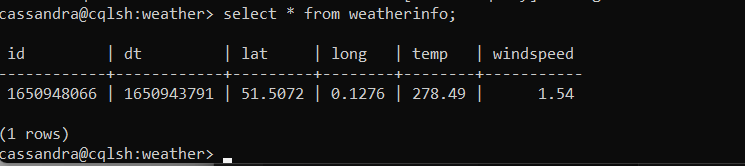
We connect and create a table through:

const cassandra = require('cassandra-driver');

const client = new cassandra.Client({ contactPoints: ["10.4.2.96","10.4.2.112","10.4.2.105"], localDataCenter: 'datacenter1', keyspace: 'weather', credentials: { username: 'cassandra', password: 'cassandra' }

});

After connection Api fetch data and insert into weatherinfo table



The Open Weather API to fetch current weather:

const express = require('express')

const app = express()

const axios = require('axios')

const cassandra = require('cassandra-driver');

const bodyParser=require('body-parser');

const urlencoded = require('body-parser/lib/types/urlencoded');

app.use(express.json());

app.use(bodyParser.urlencoded({ extended: true}));

const client = new cassandra.Client({

    contactPoints: ["192.168.1.222","192.168.1.220","192.168.1.223"],

    localDataCenter: 'datacenter1',

    keyspace: 'weather',

    credentials: { username: 'cassandra', password: 'cassandra' }

});

app.get('/', function (req, res) {

    const query = `

    CREATE TABLE emp(

        emp\_id int PRIMARY KEY,

        emp\_name text,

        emp\_city text,

        emp\_sal varint,

        emp\_phone varint

        );

    `;

    client.execute(query, [ ])

      .then(result => console.log(result));

});

app.post('/add-report', async(req, res)=>{

    let lat= req.body.lat;

    let long= req.body.long;

    console.log(req.body);

    let body = {}

    axios.get(`https://api.openweathermap.org/data/2.5/onecall?lat=${lat}&lon=${long}&exclude=minutely,hourly,daily&appid=4fc57c093b1962e1f5db11a8b893119c`)

    .then((response)=>{

        let data = response.data;

        body["id"] = data["current"]["sunrise"]

        body["lat"] = data["lat"]

        body["long"] = data["lon"]

        body["dt"] = data["current"]["dt"]

        body["temp"] = data["current"]["temp"]

        body["windspeed"] = data["current"]["wind\_speed"]

        const query = `

        insert into weather.weatherinfo(

            id, dt, lat, long, temp, windspeed

        ) values(

            '${String(body['id'])}', '${String(body['dt'])}', '${String(lat)}', '${String(long)}', '${String(body['temp'])}', '${String(body['windspeed'])}'

        );

        `;

        client.execute(query, [ ])

        .then(result => console.log(result));

        return res.status(201).json({

            message: "Added Successfully!"

        })

    })

    .catch((err)=>{

        console.log(err)

        return res.status(500).json({

            message: "Some Error Occurred!"

        })

    });

})

app.get('/all-reports', async(req, res)=>{

    let theRequired = []

    const query = `

        select \* from weather.weatherinfo;

        `;

    client.execute(query, [ ])

    .then(result => {

        return res.json(result.rows)

    }).catch(err=>{

        console.log(err.message);

    });

})

app.listen(3000,()=>{

    console.log("server is running on port 3000");

})

POSTMAN REQUESTS:

