

Apache Cassandra 3.11.17 Installation on Windows OS

Table of Contents

1. Overview:	4
2. Prerequisites:	5
3. Install Cassandra:	6
3.1. Download Cassandra Binaries:	6
3.2. Set up Environment variables:	8
3.3. Verify Cassandra Installation:	12
4. Configure Cassandra:	14
5. Start Cassandra:	16
5.1. Start Cassandra Service:	16
5.2. Verify Node Status:	17
6. Configure Multi-Node Cluster:	19
6.1. Extract Cassandra Binaries:	19
6.2. Modify Configuration Files:	19
6.3. Remove Environment Variable:	22
6.4. Start Cassandra Service:	23
6.5. Verify Node Status:	25
7. CQLSH:	26
7.1. Describe Cluster:	28
7.2. View Version:	28
7.3. Show Host:	28
7.4. Create Keyspace:	29
7.5. Create Table:	30
7.6. Insert Data:	31
7.7. Read Data:	31
7.8. Batch Queries:	32
7.9. Data Replication:	33
8. NodeTool Utility:	34
8.1. Nodetool help:	35
8.2. Nodetool describecluster:	35

8.3.	Nodetool status:	36
8.4.	Nodetool ring:	36
8.5.	Nodetool gossipinfo:	37
8.6.	Nodetool getlogginglevels:	37
8.7.	Nodetool netstats:	38
8.8.	Nodetool tablestats:	38
8.9.	Nodetool cleanup:	38
8.10.	Nodetool compact:	38
8.11.	Nodetool decommission:	39
8.12.	Nodetool drain:	39
8.13.	Nodetool garbagecollect:	39
8.14.	Nodetool join:	40
8.15.	Nodetool removemode:	40
8.16.	Nodetool assassinate:	40
8.17.	Nodetool repair:	41

This document outlines the steps needed to install **Apache Cassandra 3.11.17** with single-node and multi-node cluster configuration on Windows Operating system.

1. Overview:

Apache Cassandra is an open-source, highly scalable and distributed NoSQL (Not only SQL) database management system that is designed to handle large amounts of structured and unstructured data distributed across multiple commodity servers.

Apache Cassandra is initially developed at **Facebook** with a combination of distribution storage and replication model from Amazon Dynamo and column-family data model from Google Bigtable. It has a distributed architecture where large volumes of structured and unstructured data can be stored on multiple commodity hardware machines which provides high availability with no single point of failure, making it a popular choice for enterprises needing robust data solutions.

Apache Cassandra follows decentralized, peer to peer model. Unlike traditional databases that use master-slave architecture, Cassandra operates on peer-to-peer model which is nothing but all nodes in Cassandra cluster are identical, with no master nodes and each node communicates with other nodes directly, ensuring no bottlenecks and single point of failure.

The key components of Cassandra include **Node, Data Center, Cluster, Commit log, Mem-table, SSTable**. It is important to understand these terminologies in Cassandra.

- **Node:** It is the basic component in Cassandra. A node is nothing but a machine where data is actually stored
- **Data Center:** It is a collection of many nodes.
- **Cluster:** It is a collection of multiple data centers
- **Commit Log:** It is the first entry point while data to Cassandra. Data is first written to commit log which is a durable write-ahead log on disk. This helps in data recovery and sync in case of any crash in nodes.
- **Mem-table:** After writing data to commit log, data is stored in memtable which is an in-memory data structure.
- **SSTable:** Once the mem-table reaches a certain size, then data is flushed to disk and is stored in a SSTable (Sorted String table), an immutable data file.
- **Consistency levels:** Cassandra allows users to choose consistency level for their read and write operations, balancing between consistency and availability.
- **Partitioning:** Cassandra uses partitioning concept to distribute data across cluster. It hashes partition key of a row with a consistent hashing mechanism and determines which node will store that row.
- **Replication Strategy:** Based on the replication factor configured, Cassandra replicates partitions across multiple nodes to ensure data availability and fault tolerance. Cassandra

follows two replication strategies – **SimpleStrategy** (allows to specify single replication factor to copy data across cluster) **NetworkTopologyStrategy** (allows to specify replication factor for each data center in the cluster)

Note:

The latest release of Apache Cassandra from 4.0+ does not support the direct installation on Windows operating system. If you would like to use Apache Cassandra 4.0+ version, then go with Docker image or install it on Linux operating system.

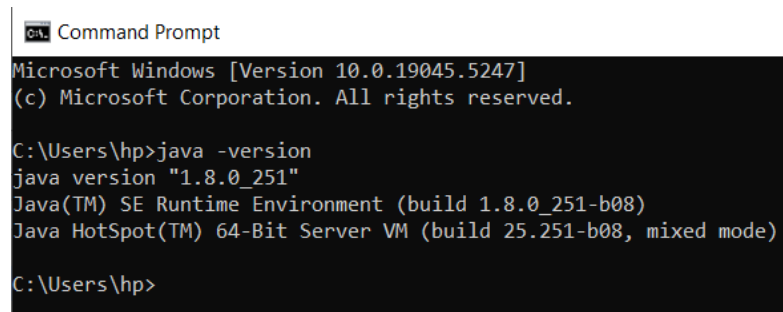
2. Prerequisites:

The following prerequisites need to be installed before running Cassandra.

1. **File Archiver:** Any file archiver such as **7zip** or **WinRAR** is needed to unzip the downloaded Spark binaries. 7zip can be downloaded from the [7zip Downloads](#) website and WinRAR can be downloaded from the [RAR lab Downloads](#) website.
2. **JRE 8:** Cassandra 3.11 requires Java 8 runtime environment supporting up to JDK 1.8_251 release. You can download the JDK 8u251 release from [Oracle Java Archive Downloads](#) website. For the complete JDK installation steps, look at [here](#).

Verify the installed Java version using the below command:

```
java -version
```



```
Command Prompt
Microsoft Windows [Version 10.0.19045.5247]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>java -version
java version "1.8.0_251"
Java(TM) SE Runtime Environment (build 1.8.0_251-b08)
Java HotSpot(TM) 64-Bit Server VM (build 25.251-b08, mixed mode)

C:\Users\hp>
```

3. **Python 2.7:** Cassandra 3.11 requires Python 2.7 to be installed to be able to use **cqlsh** tool. You can install Python 2.7.8 release from the official [Python Downloads](#) website and set the python install location in PATH environment variable.

Verify the installed Python version using the below command:

```
python --version
```

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.5247]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>python --version
Python 2.7.8

C:\Users\hp>
```

It displays that **Python 2.7.8** version is installed.

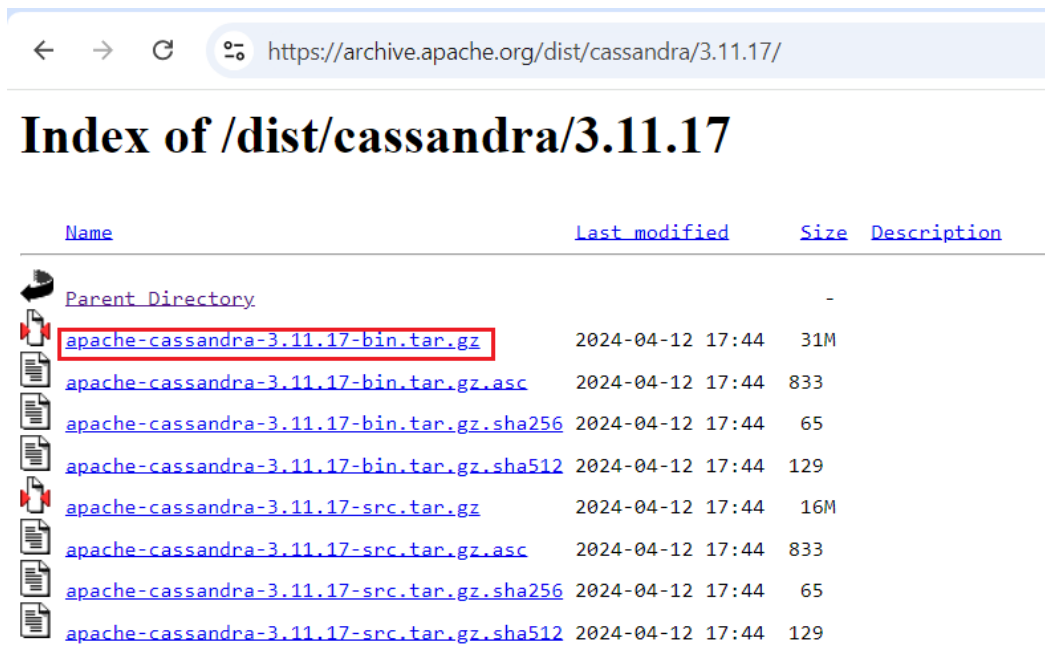
3. Install Cassandra:

Let us see the step by step procedure to install Apache Cassandra in Windows.

3.1. Download Cassandra Binaries:

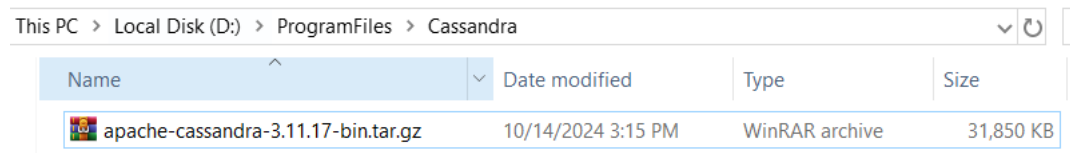
You can get the stable release from the official [Apache Cassandra Downloads](https://cassandra.apache.org/) website. At the time of this document preparation, the most recent stable release is **4.1.7** which cannot be used on Windows system directly. So, let us go the [Cassandra Archive location](https://archive.apache.org/dist/cassandra/3.11.17/) and download **3.11.17** release.

On the [Cassandra Archive 3.11.17](https://archive.apache.org/dist/cassandra/3.11.17/) page, click on [apache-cassandra-3.11.17-bin.tar.gz](https://archive.apache.org/dist/cassandra/3.11.17/bin.tar.gz) link which downloads the file into your **Downloads** folder in your machine.

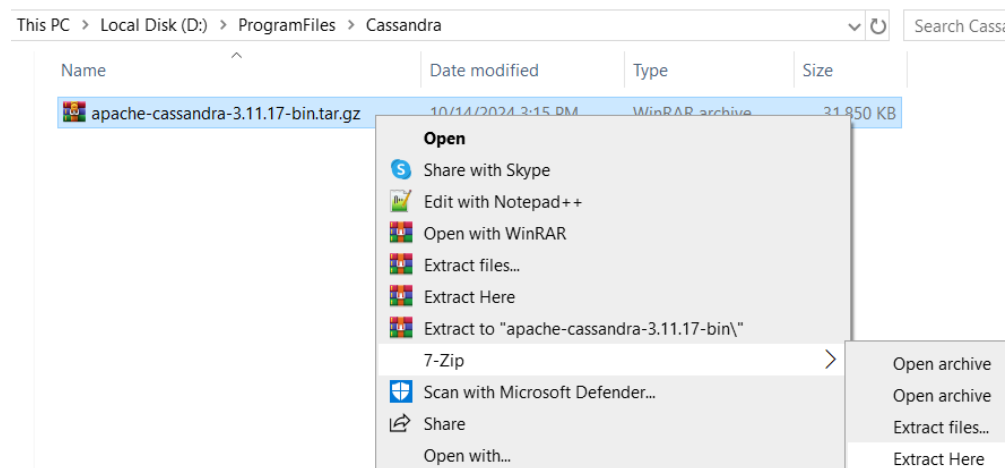


After the binary file is downloaded, unpack it using any file archiver (**7zip** or **WinRAR**) utility as below:

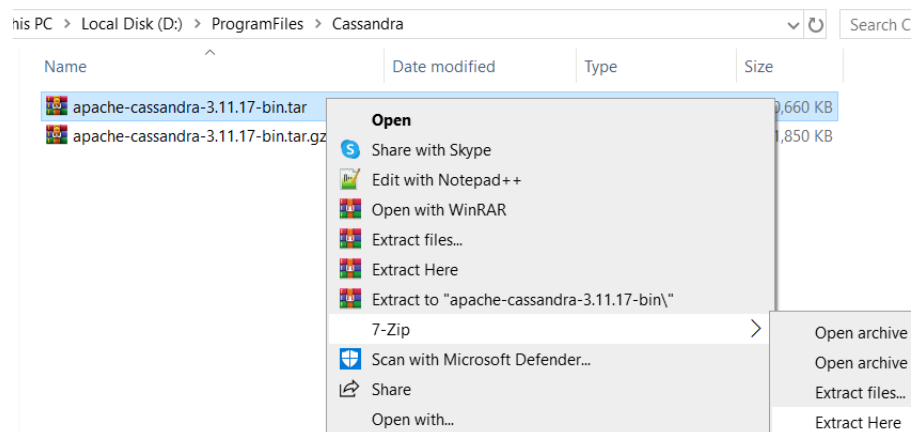
- Choose the installation directory in your machine and copy `apache-cassandra-3.11.17-bin.tar.gz` file to that directory. Here, we are choosing Cassandra installation directory as `D:\ProgramFiles\Cassandra`.



- Right click on `apache-cassandra-3.11.17-bin.tar.gz` and choose **7-Zip -> Extract Here** option which extracts a new packed file `apache-cassandra-3.11.17-bin.tar`.



- Next, unpack `apache-cassandra-3.11.17-bin.tar` file using **7zip** utility.



- The tar file extraction may take few minutes to finish. After finishing, you see a folder named `apache-cassandra-3.11.17-bin` which consists of Cassandra binaries and libraries.

his PC > Local Disk (D:) > ProgramFiles > Cassandra > apache-cassandra-3.11.17

Name	Date modified	Type	Size
bin	4/12/2024 11:10 PM	File folder	
conf	4/12/2024 11:10 PM	File folder	
doc	4/12/2024 11:10 PM	File folder	
interface	4/12/2024 11:10 PM	File folder	
lib	4/12/2024 11:10 PM	File folder	
pylib	4/12/2024 11:10 PM	File folder	
tools	4/12/2024 11:10 PM	File folder	
CASSANDRA-14092.txt	4/12/2024 11:10 PM	Text Document	5 KB
CHANGES.txt	4/12/2024 11:10 PM	Text Document	386 KB
LICENSE.txt	4/12/2024 11:10 PM	Text Document	12 KB
NEWS.txt	4/12/2024 11:10 PM	Text Document	118 KB
NOTICE.txt	4/12/2024 11:10 PM	Text Document	3 KB

3.2. Set up Environment variables:

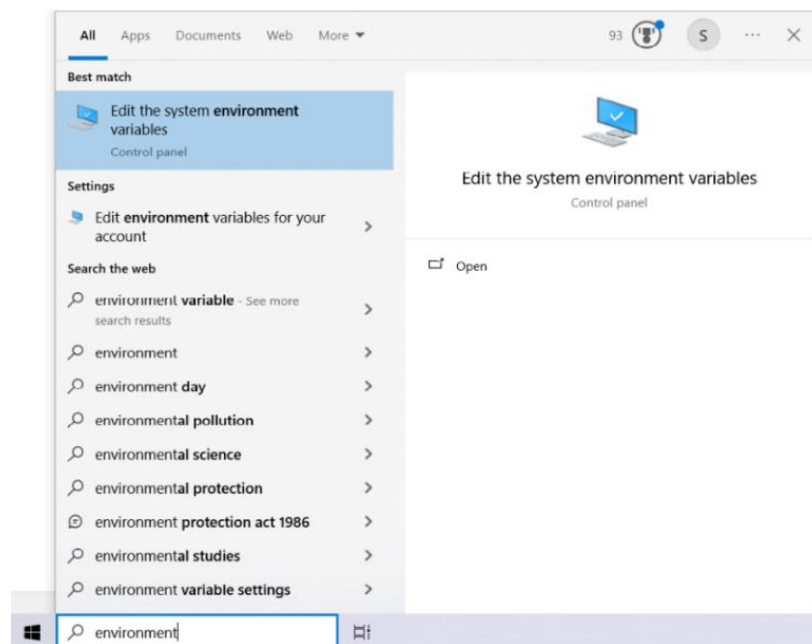
After installing pre-requisites and Cassandra binaries, we should configure the below environment variables defining Java and Cassandra default paths.

- **JAVA_HOME:** This is the JDK installation directory path in the machine (*in my machine, it is `D:\ProgramFiles\Java\jdk1.8.0_251`*). Ignore it if this is already done.
- **CASSANDRA_HOME:** This is the Cassandra installation directory path in the machine (*in our case, it is `D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17`*)

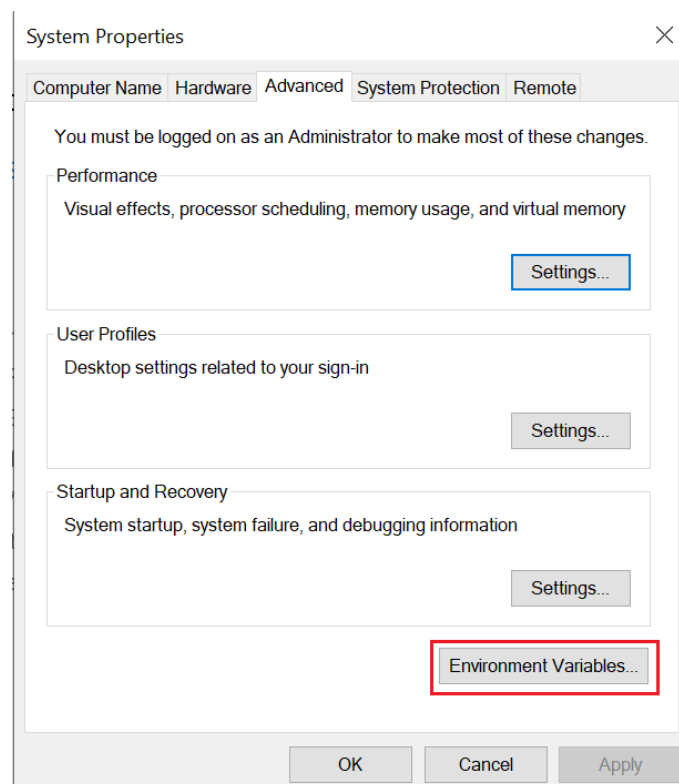
These variables need to be added to either **User environment variables** or **System environment variables** depending on Cassandra configuration needed for a single user or for multiple users.

In this tutorial, we will add User environment variables since we are configuring Cassandra for a single user. If you would like to configure Cassandra for multiple users, then define System environment variables.

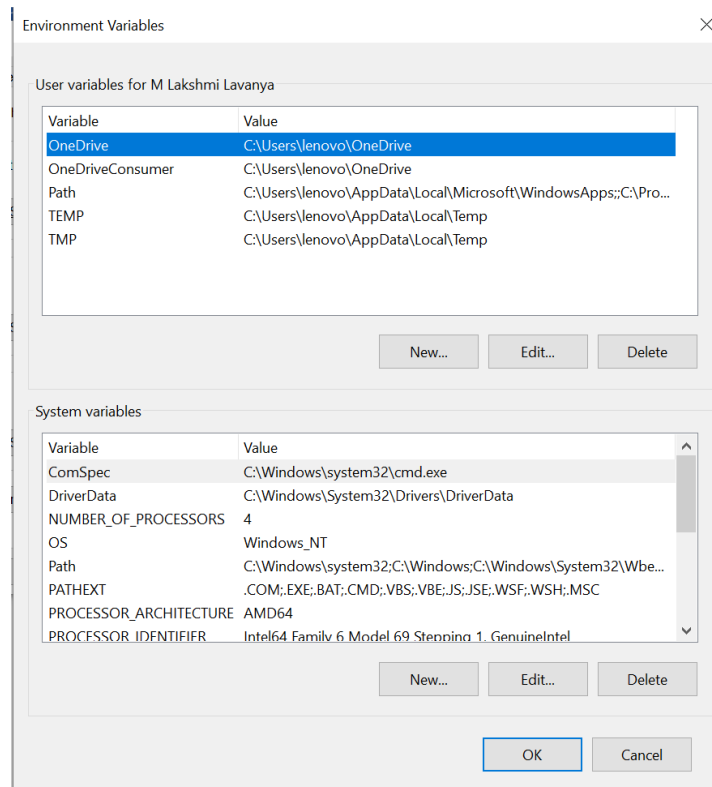
In the Windows search bar, start typing “environment variables” and select the first match which opens up **System Properties** dialog.



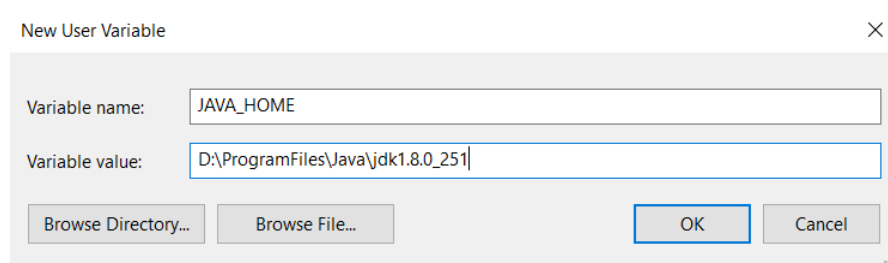
On the **System Properties** window, press **Environment Variables** button.



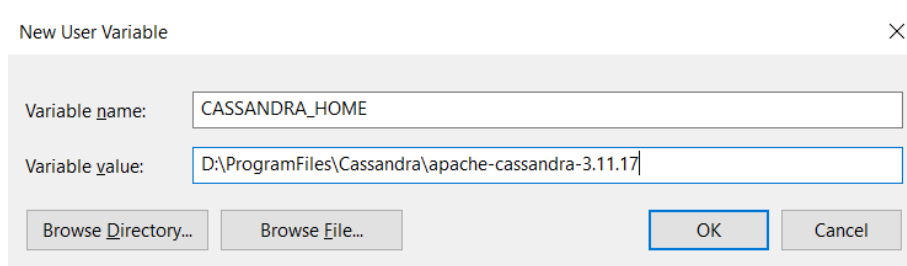
In the **Environment Variables** dialog, click on **New** under **User variables** section.



Add `JAVA_HOME` variable and press OK.

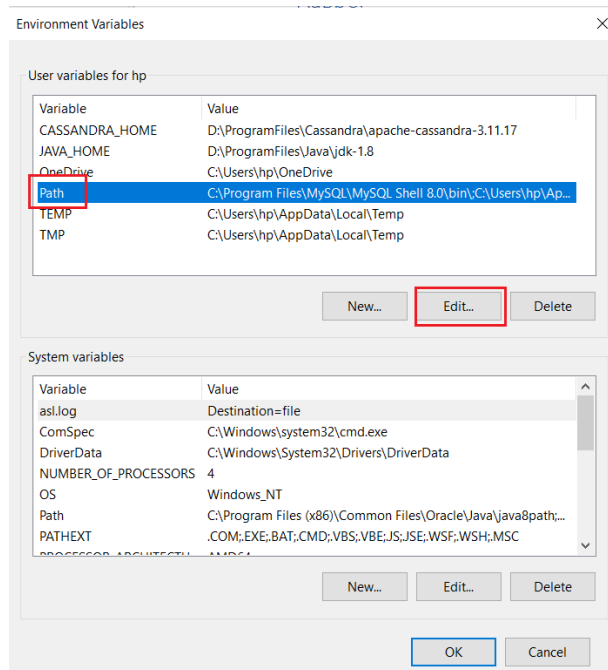


Click on **New** again and add `CASSANDRA_HOME` variable and press OK.



Now, we will update PATH variable to add Java, and Cassandra binary paths.

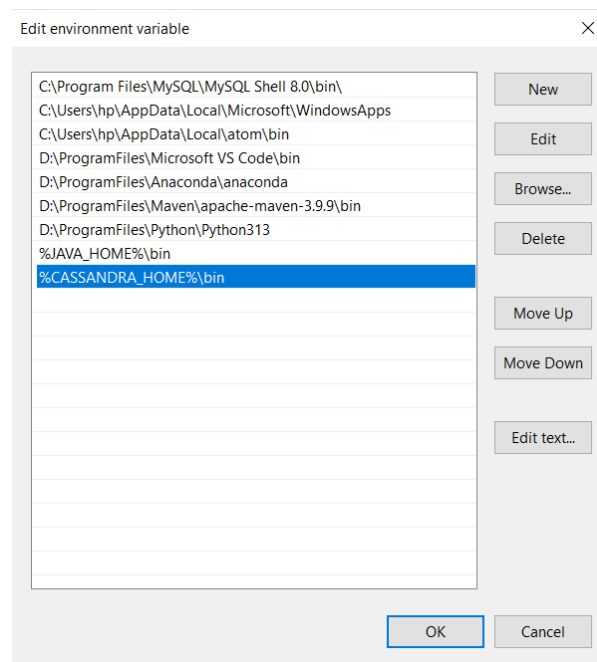
Select PATH variable under **User Variables** and press **Edit** button.



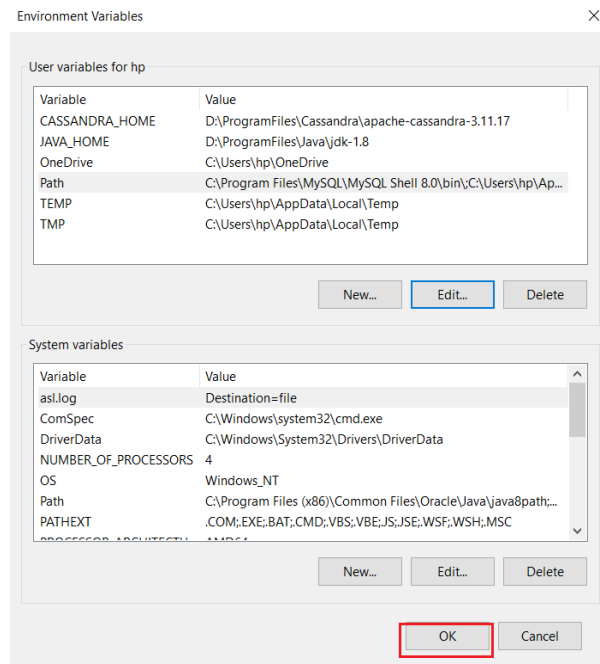
Press **New** and add the following values and press OK.

%JAVA_HOME%\bin

%CASSANDRA_HOME%\bin



Press OK again to apply environment variable changes and close window.



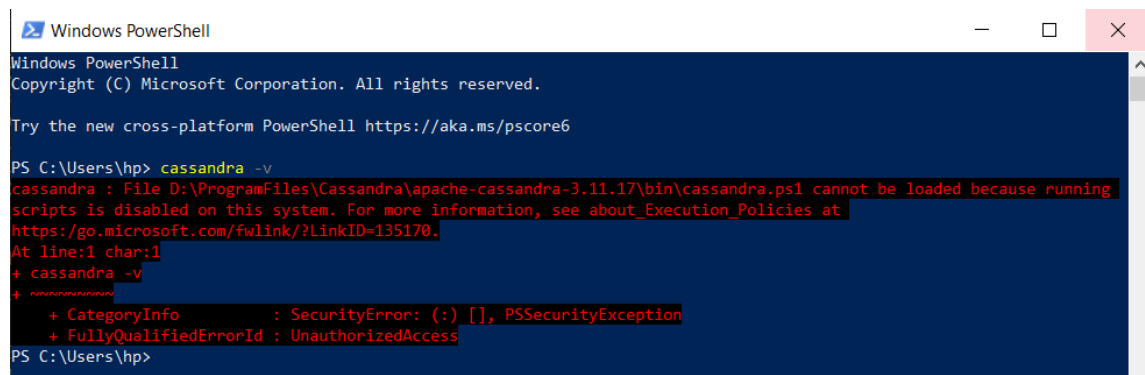
3.3. Verify Cassandra Installation:

Open **Windows PowerShell** and run the following command to verify if Cassandra is installed properly: (you cannot do this from **Command Prompt** and must use **Windows PowerShell**)

```
cassandra -v
```

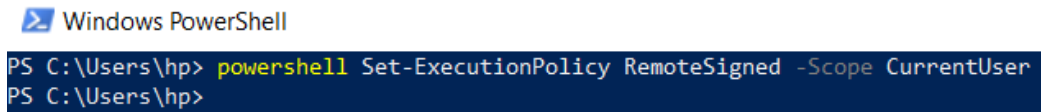
Note:

While executing the above script, you may encounter an error *cassandra : File D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin\cassandra.ps1 cannot be loaded because running scripts is disabled on this system*, as shown below. This error occurs when you do not have permissions to execute the PowerShell script.



To resolve the above error, run the below command to set the execution policy to 'Remotesigned' for the current user.

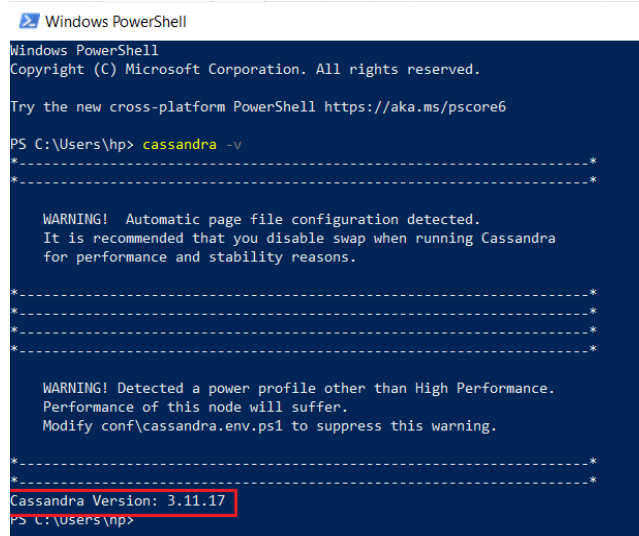
```
powershell Set-ExecutionPolicy RemoteSigned -Scope CurrentUser
```



```
Windows PowerShell
PS C:\Users\hp> powershell Set-ExecutionPolicy RemoteSigned -Scope CurrentUser
PS C:\Users\hp>
```

Now, you should be able to verify the Cassandra version installed by running this command

```
cassandra -v
```



```
Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

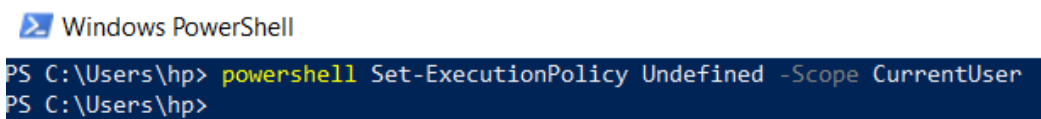
PS C:\Users\hp> cassandra -v
*-----*
*-----*
WARNING! Automatic page file configuration detected.
It is recommended that you disable swap when running Cassandra
for performance and stability reasons.
*-----*
*-----*
*-----*
WARNING! Detected a power profile other than High Performance.
Performance of this node will suffer.
Modify conf\cassandra.env.ps1 to suppress this warning.
*-----*
*-----*
Cassandra Version: 3.11.17
PS C:\Users\hp>
```

Here, it shows **3.11.17** version which indicates that Cassandra has been installed successfully.

Note:

It is necessary to have PowerShell execution policy set to either 'RemoteSigned' or 'Unrestricted' to be able to run Cassandra commands from **Windows PowerShell** prompt in Windows system. Going forward, we will execute Cassandra commands from **Command Prompt** itself and so let us revert the execution policy that we have set earlier by executing the below command in Windows PowerShell.

```
powershell Set-ExecutionPolicy Undefined -Scope CurrentUser
```



```
Windows PowerShell
PS C:\Users\hp> powershell Set-ExecutionPolicy Undefined -Scope CurrentUser
PS C:\Users\hp>
```

4. Configure Cassandra:

When Cassandra is used on a single node, there is no need to make any changes to the default configuration but if you would like to setup a Cassandra cluster with multiple nodes, it is important to know the configuration details.

The key configuration files used in Cassandra are available under `%CASSANDRA_HOME%\conf` folder.

1. **cassandra.yml** – It is the primary config file for each node instance to configure the cluster name, IP address, directory paths, etc.
2. **cassandra-env.sh** – Environment file to set java environment settings such as `MAX_HEAP_SIZE`.
3. **Cassandra-rackdc.properties** – Config file used to set the rack and data center to which the node belongs to.
4. **Cassandra-topology.properties** – Config file used to specify the IP address for racks and data centers in the cluster.

Though we are using Cassandra with single node, let us see how some configuration changes can be made for cluster setup.

Open `cassandra.yml` from `%CASSANDRA_HOME%\conf` directory and set the following properties:

- `cluster_name`: Name of the cluster. By default, it is set to `Test Cluster`. You can change it to the desired name, for instance, `CassandraDBCluster`. Make sure that all nodes in the cluster use the same `cluster_name`.
- `num_tokens`: Number of tokens. By default, this is set to 256. Any value greater than 1 is treated as virtual node so that token distribution will happen automatically.
- `listen_address`: IP address or hostname of the current node to be used by other nodes to connect to this node. By default, it is set to `localhost` but you can change it to the internal IP address of the node.
- `rpc_address`: Internal IP address for thrift client connections. By default, it is set to `localhost` but you can change it to the internal IP address of the node.
- `seed_provider`: List of internal IP addresses of hosts that are deemed contact points. In the config file of every node, we should specify the IP address of seed nodes. By default, it is set to `"127.0.0.1"` and should be updated with internal IP addresses of all hosts in the cluster.
- `endpoint_snitch`: It gives some information about network topology so that Cassandra can efficiently route requests. By default, it is set to `SimpleSnitch` class which is appropriate for cluster setup with single datacenter. For multi-data center

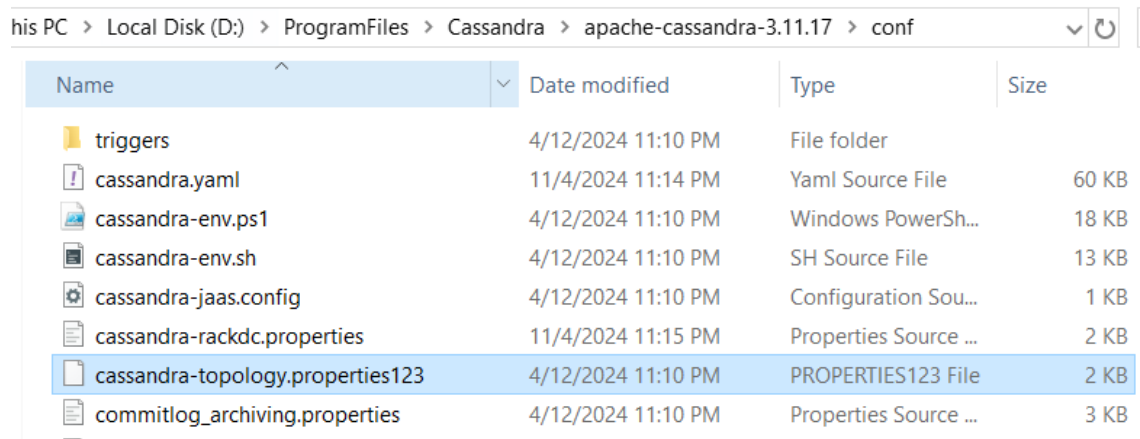
deployments, it is recommended to use `GossipingPropertyFileSnitch` with which rack and datacenter for the local node are specified in `Cassandra-rackdc.properties` and propagated to other nodes via gossip. Cassandra supports other types of snitches such as `PropertyFileSnitch`, `Ec2Snitch`, `Ec2MultiRegionSnitch`, `RackInferringSnitch`, etc.

- `auto_bootstrap`: This configuration is not available in the file and can be added and set to `false`. This configuration is optional if nodes are added to existing cluster but required when creating a new cluster with no data.

Locate the above settings in `cassandra.yml` file and modify as below:

```
cluster_name: 'CassandraDBCluster'
listen_address: 127.0.0.1
rpc_address: 127.0.0.1
seed_provider:
  - class_name: org.apache.cassandra.locator.SimpleSeedProvider
    parameters:
      - seeds: "127.0.0.1"
endpoint_snitch: GossipingPropertyFileSnitch
auto_bootstrap: false
```

When the `endpoint_snitch` is set to `GossipingPropertyFileSnitch` in `cassandra.yml` file, it always loads `cassandra-topology.properties` file if present and so, it is recommended to remove or rename this file in `%CASSANDRA_HOME%\conf` directory on all nodes.



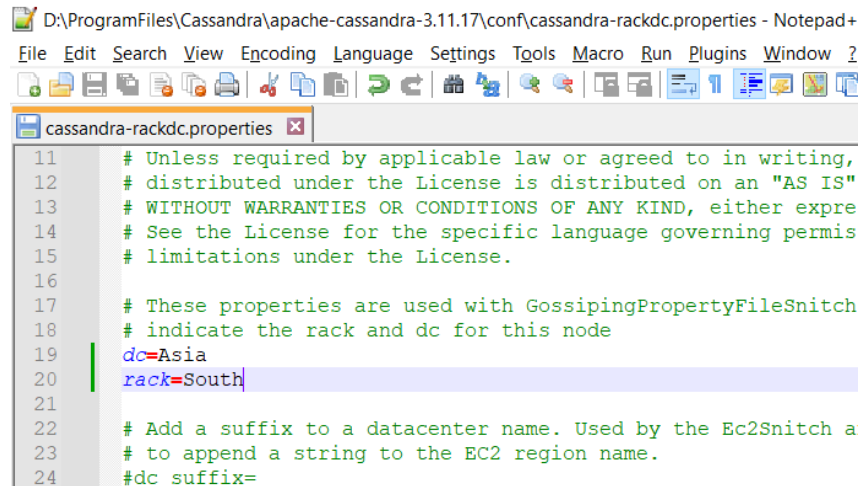
his PC > Local Disk (D:) > ProgramFiles > Cassandra > apache-cassandra-3.11.17 > conf

Name	Date modified	Type	Size
triggers	4/12/2024 11:10 PM	File folder	
cassandra.yml	11/4/2024 11:14 PM	Yaml Source File	60 KB
cassandra-env.ps1	4/12/2024 11:10 PM	Windows PowerSh...	18 KB
cassandra-env.sh	4/12/2024 11:10 PM	SH Source File	13 KB
cassandra-jaas.config	4/12/2024 11:10 PM	Configuration Sou...	1 KB
cassandra-rackdc.properties	11/4/2024 11:15 PM	Properties Source ...	2 KB
cassandra-topology.properties123	4/12/2024 11:10 PM	PROPERTIES123 File	2 KB
commitlog_archiving.properties	4/12/2024 11:10 PM	Properties Source ...	3 KB

Now, we can assign a datacenter and a rack name of the current node in `%CASSANDRA_HOME%\conf\cassandra-rackdc.properties` file. By default, `dc` is set to `dc1` and `rack` is set to `rack1`. Change these settings if you want to put the node on a different rack in a different datacenter.

In `cassandra-rackdc.properties` file, modify as below:

```
dc=Asia
rack=South
```



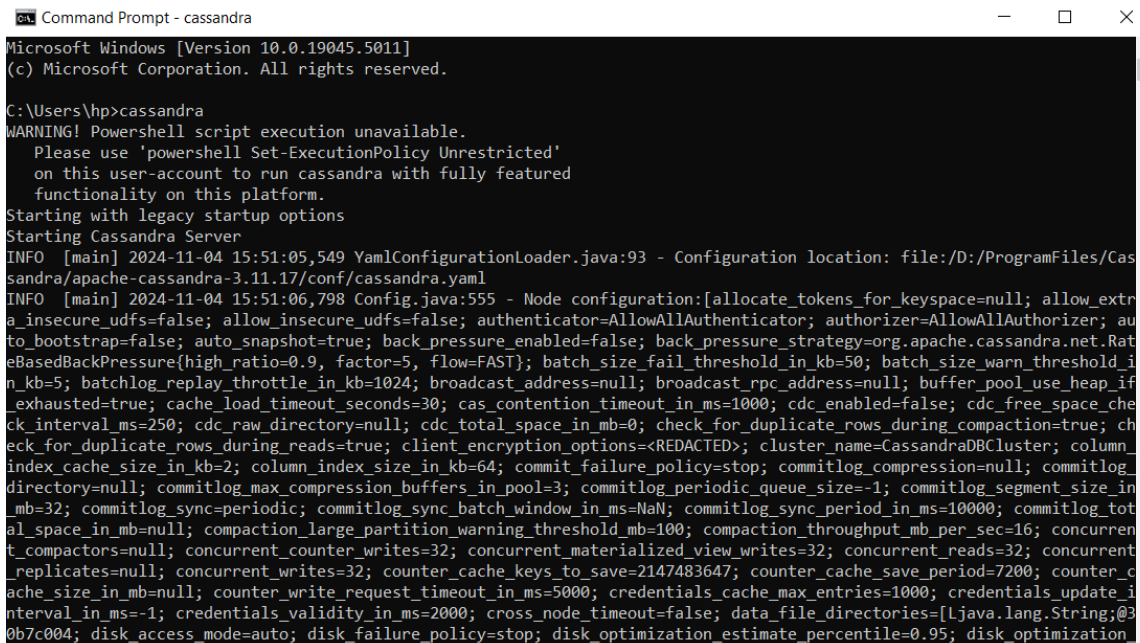
5. Start Cassandra:

It is time to start the Cassandra service and test the connection.

5.1. Start Cassandra Service:

Open **Command Prompt** and start Cassandra service using the below command:

```
cassandra
```




```

61805722479, 3940901447638997505, 4090988768067983344, 4094476459790089890, 4101194620958841001, 4135121121262872145, 41
47222441685375432, 4291127629069345553, 4299691422361713208, 4345133470886604483, 4502108715109004655, 45662032236530682
4, 4657364847329478655, 480064969698452572, 4861393139745299557, 4936637902862111774, 4951764796520171424, 5041488644686
039931, 5141251703003215490, 5234548352294079437, 5249745494224681877, 5276776363265125381, 5341067158627933453, 5356022
315289864244, 5505414569388553281, 5947320301395286806, 5964448102920499855, 5976743306459030227, 6149834389298781345, 6
16935394953573688, 6252487622068581148, 6311673945759522311, 6378328524850616581, 6421541172671519055, 65575319270630621
72, 6609561935489953240, 6751900433691548334, 6774242367089314308, 680663296013548161, 6939008319541526405, 696650518472
5587872, 6987846021164193551, 7072963785972566886, 7103469070463959878, 713828549554573561, 7553413995239462666, 7644842
59368383848, 7648436005418275591, 7700109442743567620, 7712094717719383668, 7718180553655751817, 7751088348684948084, 78
00498386799930014, 7968980211998771930, 7998686796765875243, 8153141935840144760, 8279401591116454503, 83315389368235470
71, 835765426332002881, 8426584744379278506, 8600095122214242184, 863735089149857987, 8696196755690747165, 8727187669105
365463, 8813511851530648354, 8851876625576432097, 889154596127442779, 8945536090725393954, 9125731636500548706, 91283906
66290355109, 945146485359895093, 972350692951721767]
INFO [main] 2024-11-04 15:51:21,843 StorageService.java:1670 - JOINING: Finish joining ring
INFO [main] 2024-11-04 15:51:22,476 StorageService.java:2595 - Node /127.0.0.1 state jump to NORMAL
INFO [main] 2024-11-04 15:51:22,765 NativeTransportService.java:73 - Netty using Java NIO event loop
INFO [main] 2024-11-04 15:51:22,872 Server.java:158 - Using Netty Version: [netty-buffer=netty-buffer-4.0.44.Final.4528
12a, netty-codec=netty-codec-4.0.44.Final.452812a, netty-codec-haproxy=netty-codec-haproxy-4.0.44.Final.452812a, netty-c
odec-http=netty-codec-http-4.0.44.Final.452812a, netty-codec-socks=netty-codec-socks-4.0.44.Final.452812a, netty-common=
netty-common-4.0.44.Final.452812a, netty-handler=netty-handler-4.0.44.Final.452812a, netty-tcnative=netty-tcnative-1.1.3
3.Fork26.142ecbb, netty-transport=netty-transport-4.0.44.Final.452812a, netty-transport-native-epoll=netty-transport-nat
ive-epoll-4.0.44.Final.452812a, netty-transport-rxtx=netty-transport-rxtx-4.0.44.Final.452812a, netty-transport-sctp=net
ty-transport-sctp-4.0.44.Final.452812a, netty-transport-udt=netty-transport-udt-4.0.44.Final.452812a]
INFO [main] 2024-11-04 15:51:22,872 Server.java:159 - Starting listening for CQL clients on /127.0.0.1:9042 (unencrypte
d)...
INFO [main] 2024-11-04 15:51:23,329 CassandraDaemon.java:561 - Not starting RPC server as requested. Use JMX (StorageSe
rvice->startRPCServer()) or nodetool (enablethrift) to start it
INFO [main] 2024-11-04 15:51:23,330 CassandraDaemon.java:647 - Startup complete

```

On the console, you will see a message “**Node /127.0.0.1 state jump to NORMAL**” which indicates that the Cassandra instance has started and is up and running. Do not close this command prompt window.

We can also use the `jps` command to verify Cassandra service:

```
jps
```

It should display **CassandraDaemon** which denotes that **Cassandra** service is running.

```

C:\Users\hp>jps
10132 Jps
20356 CassandraDaemon

C:\Users\hp>

```

5.2. Verify Node Status:

Now, let us check the status of current node using `nodetool` utility with the following command:

```
nodetool status
```

```
Command Prompt
Microsoft Windows [Version 10.0.19045.5247]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>nodetool status
Datacenter: Asia
=====
Status=Up/Down
// State=Normal/Leaving/Joining/Moving
-- Address      Load          Tokens      Owns (effective)  Host ID                               Rack
UN 127.0.0.1      71 KiB        256         100.0%            d0007c4c-2473-40e7-b81c-99b28d4c7e77  South

C:\Users\hp>
```

Here, you can see the process status, listening address, tokens, host ID, data center and rack name of the current node.

Check the token distribution of the node using the following command

```
nodetool ring
```

```
Command Prompt
C:\Users\hp>nodetool ring

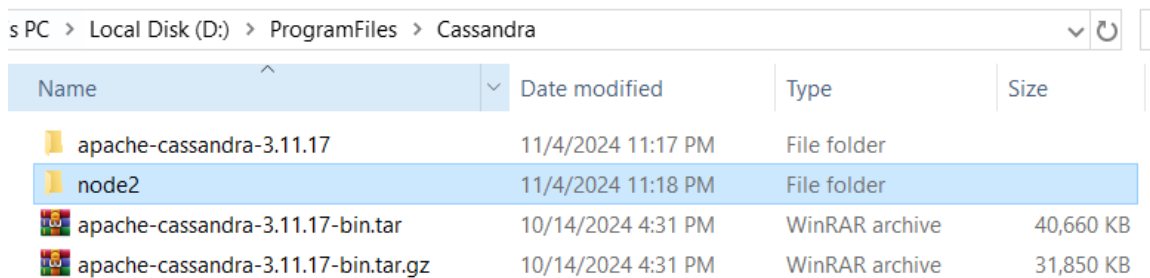
Datacenter: Asia
=====
Address      Rack      Status State  Load          Owns          Token
127.0.0.1    South     Up      Normal  71 KiB        100.00%      9198172161589035499
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -9220766189894006494
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -9196783473201380593
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -9174924999278814718
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -9082946624651449735
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8945939214799445344
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8861366292819070799
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8854521770147567993
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8761557866473840096
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8612738747689475005
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8497631400471766578
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8233610501614516548
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8219959649294129584
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8111325480103805980
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8093252645883671730
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8078908293991773158
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -8043736546566491682
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -7878559214147165878
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -7866918914852178298
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -7847196978973970820
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -7840220626997078810
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -7694745601724351458
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -7686091334425154620
127.0.0.1    South     Up      Normal  71 KiB        100.00%      -7644384978999218452
```

6. Configure Multi-Node Cluster:

For the purpose of this documentation, we will create a second Cassandra node in the same system with a different address but in the real-time project, we should configure one node per system in a Cassandra cluster.

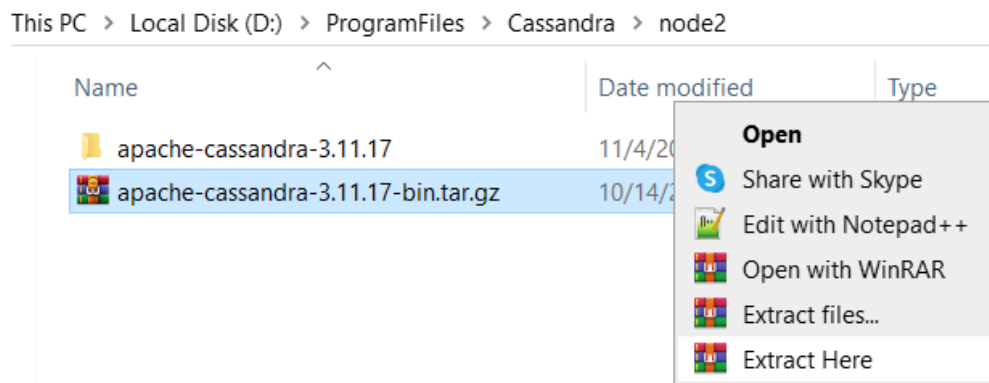
6.1. Extract Cassandra Binaries:

To set up a new Cassandra node, create a new folder where a new instance of Cassandra can be installed. In my case, I created `node2` folder under `D:\ProgramFiles\Cassandra` location.



s PC > Local Disk (D:) > ProgramFiles > Cassandra				
Name	Date modified	Type	Size	
apache-cassandra-3.11.17	11/4/2024 11:17 PM	File folder		
node2	11/4/2024 11:18 PM	File folder		
apache-cassandra-3.11.17-bin.tar	10/14/2024 4:31 PM	WinRAR archive	40,660 KB	
apache-cassandra-3.11.17-bin.tar.gz	10/14/2024 4:31 PM	WinRAR archive	31,850 KB	

Copy the Cassandra binary file `apache-cassandra-3.11.17-bin.tar.gz` to the new folder and extract contents of it.



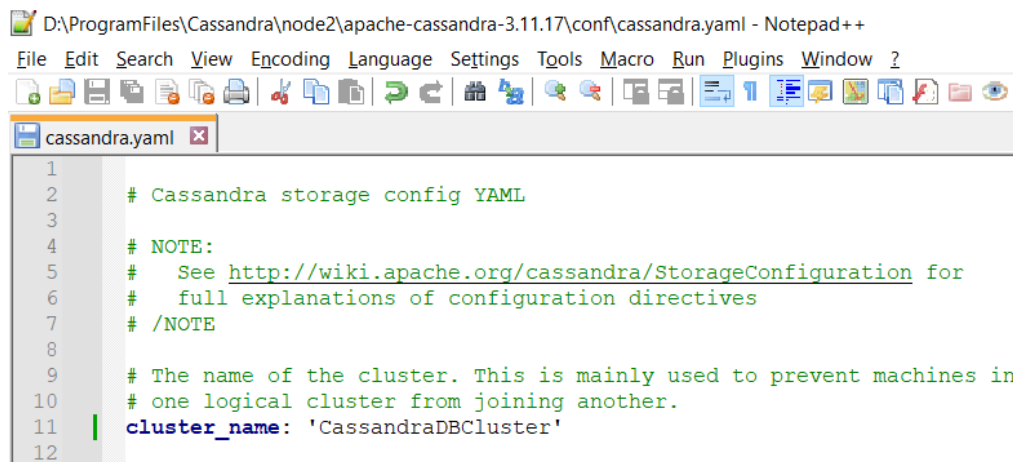
6.2. Modify Configuration Files:

We should update the following files in the new Cassandra install location to configure the second node.

```
cassandra.yaml  
cassandra-rackdc.properties  
cassandra-env.sh
```

Go to `D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\conf` and edit `cassandra.yaml` file with the following settings:

```
cluster_name: 'CassandraDBCluster'
listen_address: 127.0.0.2
rpc_address: 127.0.0.2
seed_provider:
  - class_name: org.apache.cassandra.locator.SimpleSeedProvider
    parameters:
      - seeds: "127.0.0.1,127.0.0.2"
endpoint_snitch: GossipingPropertyFileSnitch
```

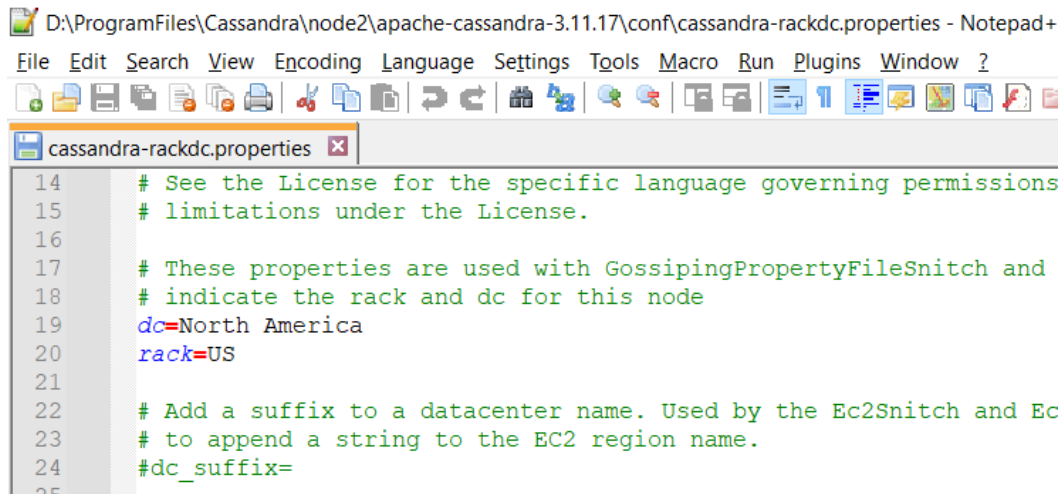


When the `endpoint_snitch` is set to `GossipingPropertyFileSnitch` in `cassandra.yaml` file, it always loads `cassandra-topology.properties` file if present and so, it is recommended to remove or rename this file.

.ocal Disk (D:) > ProgramFiles > Cassandra > node2 > apache-cassandra-3.11.17 > conf			
Name	Date modified	Type	Size
triggers	4/12/2024 11:10 PM	File folder	
cassandra.yaml	11/4/2024 11:52 PM	Yaml Source File	60 KB
cassandra-env.ps1	4/12/2024 11:10 PM	Windows PowerSh...	18 KB
cassandra-env.sh	11/4/2024 11:56 PM	SH Source File	13 KB
cassandra-jaas.config	4/12/2024 11:10 PM	Configuration Sou...	1 KB
cassandra-rackdc.properties	11/4/2024 11:50 PM	Properties Source ...	2 KB
cassandra-topology.properties123	4/12/2024 11:10 PM	PROPERTIES123 File	2 KB
commitlog_archiving.properties	4/12/2024 11:10 PM	Properties Source ...	3 KB
calhbs-sample	4/12/2024 11:10 PM	Cassandra File	7 KB

Now, open `cassandra-rackdc.properties` file and change the data center and rack name as below. Note that we are putting this new node altogether in a different data center compared to the existing node which is on Asia DC.

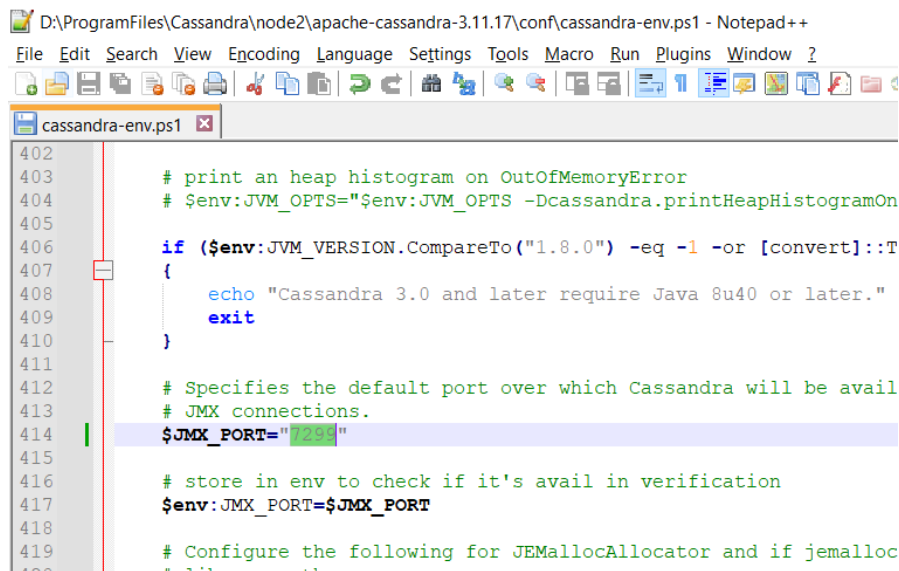
```
dc=North America
rack=US
```



```
D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\conf\cassandra-rackdc.properties - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
cassandra-rackdc.properties
14 # See the License for the specific language governing permissions
15 # limitations under the License.
16
17 # These properties are used with GossipingPropertyFileSnitch and
18 # indicate the rack and dc for this node
19 dc=North America
20 rack=US
21
22 # Add a suffix to a datacenter name. Used by the Ec2Snitch and Ec
23 # to append a string to the EC2 region name.
24 #dc_suffix=
```

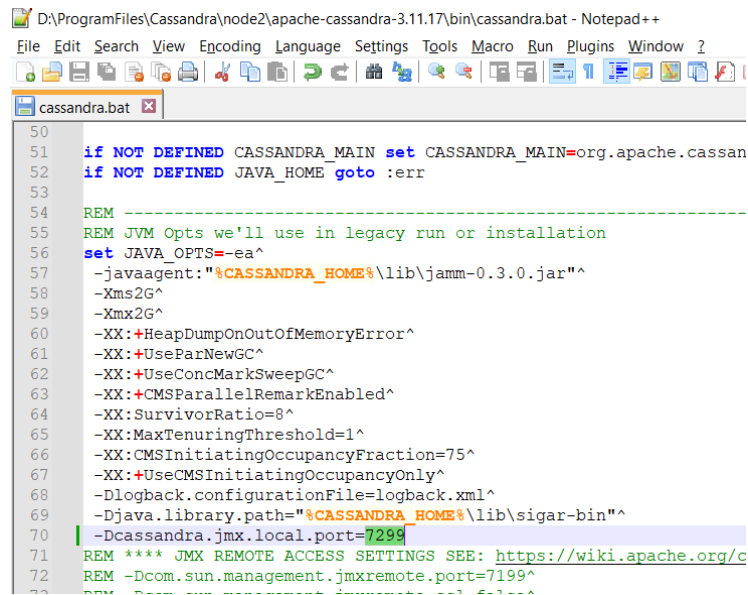
Additionally, it is required to change JMX port number in `cassandra-env.ps1` and `cassandra.bat` files.

Open `cassandra-env.ps1` in `D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\conf` location and update `JMX_PORT` value from 7199 to 7299.



```
D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\conf\cassandra-env.ps1 - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
cassandra-env.ps1
402
403 # print an heap histogram on OutOfMemoryError
404 # $env:JVM_OPTS="$env:JVM_OPTS -Dcassandra.printHeapHistogramOn
405
406 if ($env:JVM_VERSION.CompareTo("1.8.0") -eq -1 -or [convert]::T
407 {
408     echo "Cassandra 3.0 and later require Java 8u40 or later."
409     exit
410 }
411
412 # Specifies the default port over which Cassandra will be avail
413 # JMX connections.
414 $JMX_PORT='7299'
415
416 # store in env to check if it's avail in verification
417 $env:JMX_PORT=$JMX_PORT
418
419 # Configure the following for JEMallocAllocator and if jemalloc
420 # library path
```

Open `cassandra.bat` in `D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\bin` location and update `Dcassandra.jmx.local.port` value from 7199 to 7299.



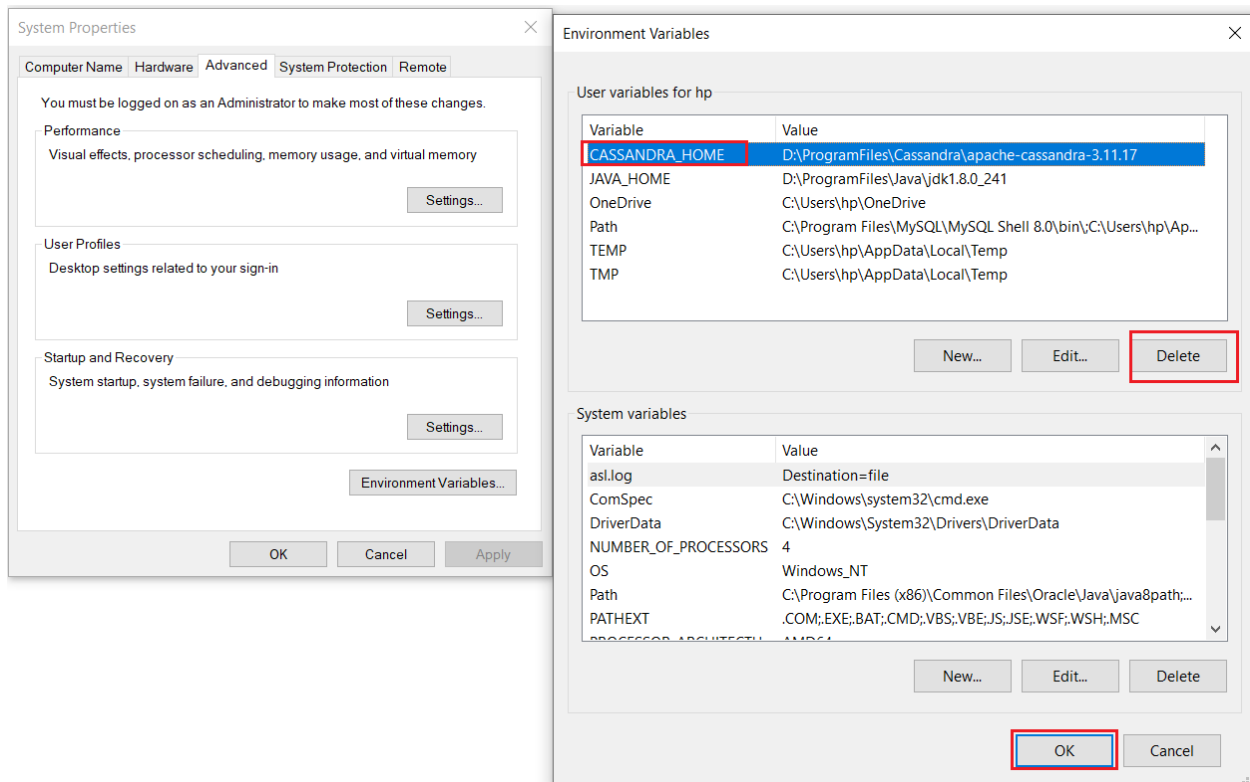
```
D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\bin\cassandra.bat - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
cassandra.bat
50
51 if NOT DEFINED CASSANDRA_MAIN set CASSANDRA_MAIN=org.apache.cassan
52 if NOT DEFINED JAVA_HOME goto :err
53
54 REM -----
55 REM JVM Opts we'll use in legacy run or installation
56 set JAVA_OPTS=-ea^
57 -javaagent:"%CASSANDRA_HOME%\lib\jamm-0.3.0.jar"^
58 -Xms2G^
59 -Xmx2G^
60 -XX:+HeapDumpOnOutOfMemoryError^
61 -XX:+UseParNewGC^
62 -XX:+UseConcMarkSweepGC^
63 -XX:+CMSParallelRemarkEnabled^
64 -XX:SurvivorRatio=8^
65 -XX:MaxTenuringThreshold=1^
66 -XX:CMSInitiatingOccupancyFraction=75^
67 -XX:+UseCMSInitiatingOccupancyOnly^
68 -Dlogback.configurationFile=logback.xml^
69 -Djava.library.path="%CASSANDRA_HOME%\lib\sigar-bin"^
70 -Dcassandra.jmx.local.port=7299
71 REM **** JMX REMOTE ACCESS SETTINGS SEE: https://wiki.apache.org/c
72 REM -Dcom.sun.management.jmxremote.port=7199^
73 REM -Dcom.sun.management.jmxremote.ssl=false^
```

6.3. Remove Environment Variable:

Whenever we try to start Cassandra, it always refers to the configuration defined in CASSANDRA_HOME environment variable and when this variable is not set, it tries to look for config files available in the current directory.

Since we are configuring the second node on the same machine, we should remove CASSANDRA_HOME environment variable that we set earlier.

Open **Environment Variables** window and select CASSANDRA_HOME under **User variables** section and click on **Delete** button. Then press OK to apply environment variable changes and close window.



6.4. Start Cassandra Service:

Open **Command Prompt** and start Cassandra service by navigating to the location where the second instance was installed:

```
D:  
cd D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\bin  
cassandra
```


It should display two **CassandraDaemon** daemons which denotes that 2 instances of **Cassandra** are running.

```
Command Prompt
Microsoft Windows [Version 10.0.19045.5011]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>jps
12664 CassandraDaemon
12540 CassandraDaemon
17292 Jps

C:\Users\hp>
```

6.5. Verify Node Status:

Now, let us check the status of nodes using `nodetool` utility by running the following commands:

```
D:
cd D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\bin
nodetool status
```

```
Command Prompt
Microsoft Windows [Version 10.0.19045.5011]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>d:

D:\>cd D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\bin

D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\bin>nodetool status
Datacenter: Asia
=====
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
-- Address      Load          Tokens       Owns (effective)  Host ID                               Rack
UN 127.0.0.1      312.38 KiB    256          100.0%            a1fb7a29-fdf0-4418-add9-517a4b35b64c  South
Datacenter: North America
-----
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
-- Address      Load          Tokens       Owns (effective)  Host ID                               Rack
UN 127.0.0.2      205.96 KiB    256          100.0%            0a7a1b59-067b-479d-aa2b-658341e27db5  US

D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\bin>
```

Here, you can see that node1 under Asia DC and node 2 under North America DC and both nodes status shows UN which means the node is Up and reporting Normal state.

Run the following `nodetool` command to get the gossip information of each node:

```
nodetool gossipinfo
```

```
D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\bin>nodetool gossipinfo
/127.0.0.1
  generation:1730746830
  heartbeat:877
  STATUS:18:NORMAL, -1088057816839888778
  LOAD:828:319881.0
  SCHEMA:14:e84b6a60-24cf-30ca-9b58-452d92911703
  DC:10:Asia
  RACK:12:South
  RELEASE_VERSION:5:3.11.17
  INTERNAL_IP:8:127.0.0.1
  RPC_ADDRESS:4:127.0.0.1
  NET_VERSION:2:11
  HOST_ID:3:a1fb7a29-fdf0-4418-add9-517a4b35b64c
  RPC_READY:21:true
  SSTABLE_VERSIONS:6:big-me
  TOKENS:17:<hidden>
/127.0.0.2
  generation:1730747023
  heartbeat:677
  STATUS:18:NORMAL, -1042386692657617965
  LOAD:643:210902.0
  SCHEMA:14:e84b6a60-24cf-30ca-9b58-452d92911703
  DC:10:North America
  RACK:12:US
  RELEASE_VERSION:5:3.11.17
  INTERNAL_IP:8:127.0.0.2
  RPC_ADDRESS:4:127.0.0.2
  NET_VERSION:2:11
  HOST_ID:3:0a7a1b59-067b-479d-aa2b-658341e27db5
  RPC_READY:32:true
  SSTABLE_VERSIONS:6:big-me
  TOKENS:17:<hidden>
```

7. CQLSH:

Cassandra provides an interactive tool called Cassandra Query Language Shell (`cqlsh`) to communicate with Cassandra database and execute queries using Cassandra Query Language (CQL). The CQL is very similar to SQL but suited for JOINless structure.

While using `cqlsh`, we can specify the IP address and port to connect to a specific Cassandra node with specific username and password. We can use the default user `cassandra` and password `cassandra` to connect to database.

Open **Command Prompt** and run the following commands to launch CQLSH tool connecting to second node in our system

```
cd D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin
cqlsh 127.0.0.2 9042
```

or

```
cd D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin
cqlsh 127.0.0.2 9042 -u cassandra -p cassandra
```

```
Command Prompt - cqlsh 127.0.0.2 9042 -u cassandra -p cassandra
Microsoft Windows [Version 10.0.19045.5011]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>d:

D:\>cd D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin

D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>cqlsh 127.0.0.2 9042 -u cassandra -p cassandra

WARNING: console codepage must be set to cp65001 to support utf-8 encoding on Windows platforms.
If you experience encoding problems, change your console codepage with 'chcp 65001' before starting cqlsh.

Connected to CassandraDBCluster at 127.0.0.2:9042.
[cqlsh 5.0.1 | Cassandra 3.11.17 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
WARNING: pyreadline dependency missing. Install to enable tab completion.
cassandra@cqlsh>
```

Here, we can see that it is connected to CassandraDBCluster at node address 127.0.0.2:9042

On `cqlsh>` prompt, use `HELP` command that lists out all possible commands that can be triggered to interact with Cassandra.

```
cqlsh> HELP

Documented shell commands:
=====
CAPTURE  CLS          COPY  DESCRIBE  EXPAND  LOGIN  SERIAL  SOURCE  UNICODE
CLEAR    CONSISTENCY  DESC  EXIT      HELP    PAGING  SHOW    TRACING

CQL help topics:
=====
AGGREGATES          CREATE_KEYSPACE          DROP_TRIGGER            TEXT
ALTER_KEYSPACE      CREATE_MATERIALIZED_VIEW DROP_TYPE                TIME
ALTER_MATERIALIZED_VIEW CREATE_ROLE              DROP_USER               TIMESTAMP
ALTER_TABLE         CREATE_TABLE             FUNCTIONS               TRUNCATE
ALTER_TYPE          CREATE_TRIGGER           GRANT                   TYPES
ALTER_USER          CREATE_TYPE              INSERT                  UPDATE
APPLY               CREATE_USER              INSERT_JSON             USE
ASCII               DATE                     INT                     UUID
BATCH               DELETE                   JSON
BEGIN              DROP_AGGREGATE           KEYWORDS
BLOB               DROP_COLUMNFAMILY       LIST_PERMISSIONS
BOOLEAN            DROP_FUNCTION            LIST_ROLES
COUNTER            DROP_INDEX              LIST_USERS
CREATE_AGGREGATE    DROP_KEYSPACE            PERMISSIONS
CREATE_COLUMNFAMILY DROP_MATERIALIZED_VIEW  REVOKE
CREATE_FUNCTION     DROP_ROLE                SELECT
CREATE_INDEX        DROP_TABLE               SELECT_JSON

cqlsh>
```

To get more help on each command, use `HELP` followed by command name. For example, to know more details about `SHOW` command, run `HELP SHOW` as shown here

```
cqlsh> HELP SHOW

SHOW [cqlsh only]

Displays information about the current cqlsh session. Can be called in
the following ways:

SHOW VERSION

Shows the version and build of the connected Cassandra instance, as
well as the versions of the CQL spec and the Thrift protocol that
the connected Cassandra instance understands.

SHOW HOST

Shows where cqlsh is currently connected.

SHOW SESSION <sessionid>

Pretty-prints the requested tracing session.

cqlsh>
```

7.1. Describe Cluster:

Use the below CQLSH command to see the information about the connected Cassandra cluster, such as the cluster name, and the partitioner and snitch in use. When you are connected to a non-system keyspace, also shows endpoint-range ownership information for the Cassandra ring.

DESCRIBE CLUSTER

```
cassandra@cqlsh>
cassandra@cqlsh> describe cluster

Cluster: CassandraDBCluster
Partitioner: Murmur3Partitioner

cassandra@cqlsh>
```

7.2. View Version:

Use the below CQLSH command to see the version and build of the connected Cassandra instance:

SHOW VERSION

```
cassandra@cqlsh> show version
[cqlsh 5.0.1 | Cassandra 3.11.17 | CQL spec 3.4.4 | Native protocol v4]
cassandra@cqlsh>
```

Here, you can see that the CQLSH version 5.0.1 and Cassandra 3.11.17 that we installed.

7.3. Show Host:

To see where the current CQLSH session is connected, run the below command:

SHOW HOST

```
cassandra@cqlsh> show host
Connected to CassandraDBCluster at 127.0.0.2:9042.
cassandra@cqlsh>
```

Here, you can see that the current CQLSH instance is connected to `CassandraDBCluster`.

7.4. Create Keyspace:

A keyspace in Cassandra is like database in RDBMS and contains column families, indexes, user defined types, data center awareness, strategy used in keyspace, replication factor, etc.

Let us create a new keyspace named `testspace` with replication factor 2 using the below command:

```
CREATE KEYSPACE testspace WITH replication =
{'class':'SimpleStrategy', 'replication_factor':2};
```

```
cassandra@cqlsh> CREATE KEYSPACE testspace WITH replication = {'class':'SimpleStrategy', 'replication_factor':2};
cassandra@cqlsh>
```

Cassandra keyspace can be created with two types of strategy declaration:

- **SimpleStrategy:** This strategy is used in the case of one data center. In this strategy, the first replica is placed on the selected node and the remaining nodes are placed in clockwise direction in the ring without considering rack or node location.
- **NetworkTopologyStrategy:** This strategy is used in the case of more than one data centers. In this strategy, you have to provide replication factor for each data center separately.

Replication Factor is another property to be specified for a keyspace. Replication factor is the number of replicas of data placed on different nodes.

Use `describe` command to verify if the new keyspace was created or not:

```
DESCRIBE KEYSPACES;
```

```
cassandra@cqlsh> DESCRIBE KEYSPACES;
system_schema system_auth system system_distributed testspace system_traces
cassandra@cqlsh>
```

The `use` command in Cassandra sets the current working keyspace. Run the following command to set `testspace` as our working keyspace:

```
USE testspace;
```

```
cassandra@cqlsh> USE testspace;  
cassandra@cqlsh:testspace>
```

7.5. Create Table:

In Cassandra, `CREATE TABLE` command is used to create a table with column families to store data just like table in RDBMS. This command expects the table name, column name, data type and a primary key. The datatypes in Cassandra can be `text`, `int`, `date`, `decimal`, `double`, `float`, `set`, `list`, `map`, etc.

Use this command to create table named `student`:

```
CREATE TABLE student (  
  student_id INT PRIMARY KEY,  
  student_name TEXT,  
  student_city TEXT,  
  student_fees INT  
);
```

```
cassandra@cqlsh:testspace> CREATE TABLE student (  
    ... student_id INT PRIMARY KEY,  
    ... student_name TEXT,  
    ... student_city TEXT,  
    ... student_fees INT  
    ... );  
cassandra@cqlsh:testspace>
```

Use `describe` command to verify if the table was created or not:

```
DESCRIBE TABLES;
```

```
cassandra@cqlsh:testspace> DESCRIBE TABLES;  
student  
cassandra@cqlsh:testspace>
```

7.6. Insert Data:

The `INSERT` command is used to insert data into table columns.

Run the following commands to insert some data into `student` table:

```
INSERT INTO student (student_id, student_name, student_city,
student_fees)
VALUES (1, 'Rakesh', 'Hyderabad', 5000);
INSERT INTO student (student_id, student_name, student_city,
student_fees)
VALUES (2, 'Ramana', 'Bangalore', 7000);
INSERT INTO student (student_id, student_name, student_city,
student_fees)
VALUES (3, 'Kranthi', 'Chennai', 4000);
```

```
cassandra@cqlsh:testspace> INSERT INTO student (student_id, student_name, student_city, student_fees)
... VALUES (1, 'Rakesh', 'Hyderabad', 5000);
cassandra@cqlsh:testspace> INSERT INTO student (student_id, student_name, student_city, student_fees)
... VALUES (2, 'Ramana', 'Bangalore', 7000);
cassandra@cqlsh:testspace> INSERT INTO student (student_id, student_name, student_city, student_fees)
... VALUES (3, 'Kranthi', 'Chennai', 4000);
cassandra@cqlsh:testspace>
```

7.7. Read Data:

Use `SELECT` command to read data from the table.

```
SELECT * FROM student;
```

```
cassandra@cqlsh:testspace> SELECT * FROM student;

 student_id | student_city | student_fees | student_name
-----+-----+-----+-----
          1 |   Hyderabad |         5000 |      Rakesh
          2 |   Bangalore |         7000 |      Ramana
          3 |    Chennai |         4000 |    Kranthi

(3 rows)
cassandra@cqlsh:testspace>
```

```
SELECT * FROM student WHERE student_id=2;
```

```
cassandra@cqlsh:testspace> SELECT * FROM student WHERE student_id=2;

student_id | student_city | student_fees | student_name
-----+-----+-----+-----
          2 |    Bangalore |         7000 |         Ramana

(1 rows)
cassandra@cqlsh:testspace>
```

7.8. Batch Queries:

In Cassandra, the `BATCH` command is used to execute multiple DML statements such as `insert`, `update`, `delete`, etc. simultaneously. It is very useful when you have to update some column as well as delete some of the existing data.

Run the following batch operations to insert a new record, update the existing record and delete a column data.

```
BEGIN BATCH
INSERT INTO student (student_id, student_name, student_city,
student_fees) VALUES (4, 'Shivani', 'Mumbai', 10000);
UPDATE student SET student_fees = 8000 WHERE student_id = 2;
DELETE student_fees FROM student WHERE student_id=1;
APPLY BATCH;
```

```
cassandra@cqlsh:testspace> BEGIN BATCH
... INSERT INTO student (student_id, student_name, student_city, student_fees) VALUES (4, 'Shivani', 'Mumbai', 10000);
... UPDATE student SET student_fees = 8000 WHERE student_id = 2;
... DELETE student_fees FROM student WHERE student_id=1;
... APPLY BATCH;
cassandra@cqlsh:testspace>
```

Now the `BATCH` is applied, we can verify it by using `SELECT` command:

```
SELECT * FROM student;
```

```
cassandra@cqlsh:testspace> SELECT * FROM student;

student_id | student_city | student_fees | student_name
-----+-----+-----+-----
          1 |    Hyderabad |         null |         Rakesh
          2 |    Bangalore |         8000 |         Ramana
          4 |     Mumbai |        10000 |        Shivani
          3 |     Chennai |         4000 |        Kranthi

(4 rows)
cassandra@cqlsh:testspace>
```

Use `exit;` command to exit out of `cqlsh`.

7.9. Data Replication:

Now, let us connect to the first node instance to verify if the data is replicated.

```
cd D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin
cqlsh 127.0.0.1 9042 -u cassandra -p cassandra
```

```
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>cqlsh 127.0.0.1 9042 -u cassandra -p cassandra
WARNING: console codepage must be set to cp65001 to support utf-8 encoding on Windows platforms.
If you experience encoding problems, change your console codepage with 'chcp 65001' before starting cqlsh.

Connected to CassandraDBCluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.11.17 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
WARNING: pyreadline dependency missing. Install to enable tab completion.
cassandra@cqlsh>
```

Run the following commands to see the table data that we created in the second node.

```
DESCRIBE KEYSPACES;
USE testspace;
DESCRIBE TABLES;
SELECT * FROM student;
```

```
cassandra@cqlsh> DESCRIBE KEYSPACES;

system_schema  system_auth  system  testspace  system_distributed  system_traces

cassandra@cqlsh> USE testspace;
cassandra@cqlsh:testspace> SHOW TABLES;
Improper SHOW command.
cassandra@cqlsh:testspace> DESCRIBE TABLES;

student

cassandra@cqlsh:testspace> SELECT * FROM student;

 student_id | student_city | student_fees | student_name
-----+-----+-----+-----
          1 |   Hyderabad |         null |      Rakesh
          2 |   Bangalore |         8000 |      Ramana
          4 |    Mumbai |        10000 |    Shivani
          3 |    Chennai |         4000 |    Kranthi

(4 rows)
cassandra@cqlsh:testspace>
```

8. NodeTool Utility:

Cassandra provides a `nodetool` utility which is a command line interface for monitoring Cassandra cluster and performing routine database operations. This utility is commonly used to output a quick summary of the ring and its current state of general health with `status` command.

The `nodetool` utility provides commands for decommissioning a node, running repair, and moving partitioning tokens and for viewing detailed metrics for tables, server metrics, and compaction statistics, etc.

When you have multi-node cluster, you can provide the node host name and JMX port to connect:

```
nodetool -h HOSTNAME -p JMX_PORT COMMAND
```

If a username and password for RMI authentication are set explicitly in the `cassandra-env.sh` file, then you must specify credentials along with hostname/IP address and port to connect to a specific Cassandra node:

```
nodetool -h HOSTNAME -p JMX_PORT -u JMX_USERNAME -pw JMX_PASSWORD  
COMMAND
```

Open **Command Prompt** and run the following commands to get node status connecting to primary node in our system:

```
cd D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin  
nodetool -h 127.0.0.1 -p 7199 -u cassandra -pw cassandra status
```

Use the below command to connect to secondary node in our system:

```
nodetool -h 127.0.0.1 -p 7299 -u cassandra -pw cassandra status
```

```
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>nodetool -h 127.0.0.1 -p 7199 -u cassandra -pw cassandra status  
Datacenter: Asia  
=====
```

Status=Up/Down						
/ State=Normal/Leaving/Joining/Moving						
	Address	Load	Tokens	Owns (effective)	Host ID	Rack
UN	127.0.0.1	294.39 KiB	256	100.0%	a1fb7a29-fdf0-4418-add9-517a4b35b64c	South

```
Datacenter: North America  
=====
```

Status=Up/Down						
/ State=Normal/Leaving/Joining/Moving						
	Address	Load	Tokens	Owns (effective)	Host ID	Rack
UN	127.0.0.2	194.15 KiB	256	100.0%	0a7a1b59-067b-479d-aa2b-658341e27db5	US

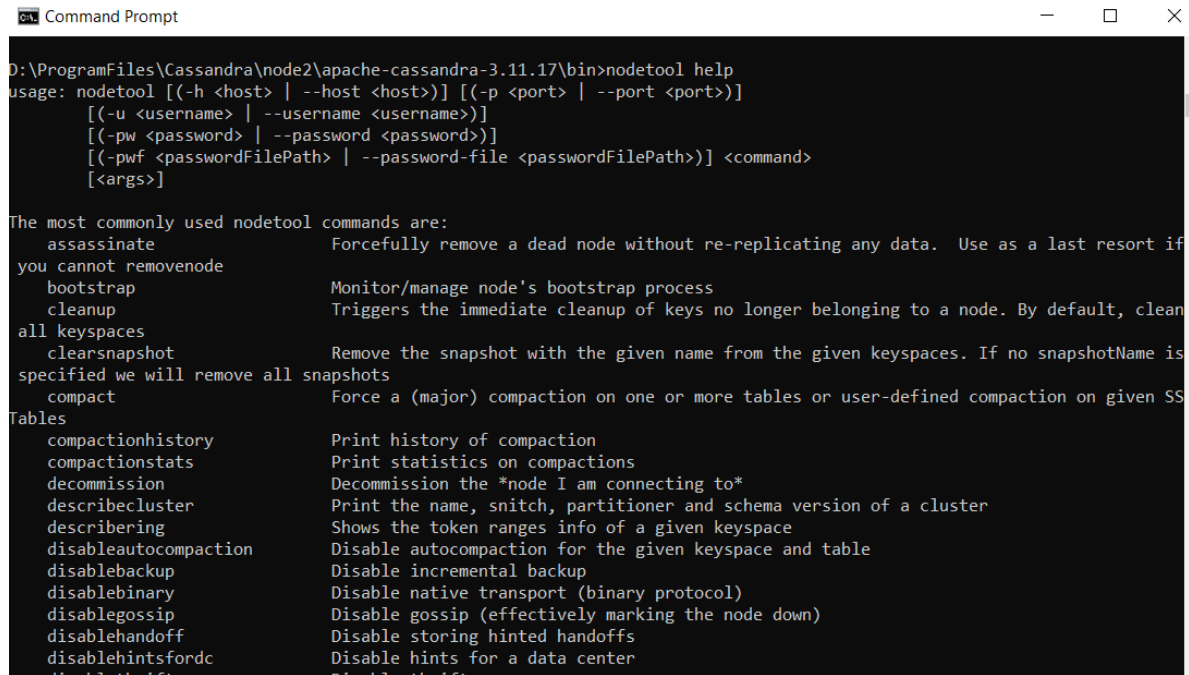
```
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>
```

Some of the generally used commands are described below for your understanding.

8.1. Nodetool help:

This is a basic command which lists all the available `nodetool` commands. This command is useful to learn about the available commands.

```
nodetool help
```



```
D:\ProgramFiles\Cassandra\node2\apache-cassandra-3.11.17\bin>nodetool help
usage: nodetool [(-h <host> | --host <host>)] [(-p <port> | --port <port>)]
      [(-u <username> | --username <username>)]
      [(-pw <password> | --password <password>)]
      [(-pwf <passwordFilePath> | --password-file <passwordFilePath>)] <command>
      [<args>]

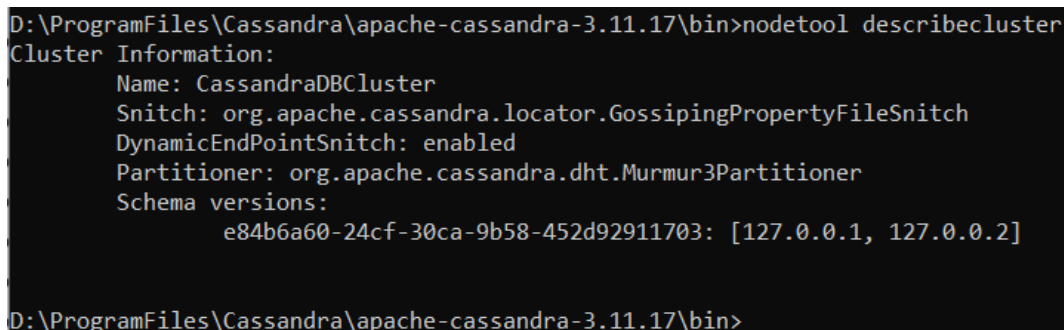
The most commonly used nodetool commands are:
  assassinate          Forcefully remove a dead node without re-replicating any data. Use as a last resort if
you cannot remove node
  bootstrap            Monitor/manage node's bootstrap process
  cleanup              Triggers the immediate cleanup of keys no longer belonging to a node. By default, clean
all keyspaces
  clearsnapshot        Remove the snapshot with the given name from the given keyspaces. If no snapshotName is
specified we will remove all snapshots
  compact              Force a (major) compaction on one or more tables or user-defined compaction on given SS
Tables
  compactionhistory    Print history of compaction
  compactionstats      Print statistics on compactions
  decommission         Decommission the *node I am connecting to*
  describecluster      Print the name, snitch, partitioner and schema version of a cluster
  describing           Shows the token ranges info of a given keyspace
  disableautocompaction Disable autocompaction for the given keyspace and table
  disablebackup        Disable incremental backup
  disablebinary        Disable native transport (binary protocol)
  disablegossip        Disable gossip (effectively marking the node down)
  disablehandoff       Disable storing hinted handoffs
  disablehintsforcdc   Disable hints for a data center
  disablethrift        Disable the thrift server
```

8.2. Nodetool describecluster:

This command provides the basic cluster information such as cluster name, endpoint-snitch being used, partitioner and schema versions.

Follow the below syntax to use this command:

```
nodetool [options] describecluster
```



```
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>nodetool describecluster
Cluster Information:
  Name: CassandraDBCluster
  Snitch: org.apache.cassandra.locator.GossipingPropertyFileSnitch
  DynamicEndPointSnitch: enabled
  Partitioner: org.apache.cassandra.dht.Murmur3Partitioner
  Schema versions:
    e84b6a60-24cf-30ca-9b58-452d92911703: [127.0.0.1, 127.0.0.2]

D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>
```

8.3. Nodetool status:

This command is useful to monitor the cluster's health and performance. It can help detect any ongoing anomalies as well as the status of each of the nodes.

Follow the below syntax to use this command:

```
nodetool <options> status ( -r | --resolve-ip ) -- <keyspace>
```

```
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>nodetool status
Datacenter: Asia
=====
Status=Up/Down
// State=Normal/Leaving/Joining/Moving
-- Address      Load          Tokens       Owns (effective)  Host ID                               Rack
UN 127.0.0.1     241.51 KiB    256          100.0%            a0eadcc0-2bb4-4b09-8ae3-4b0e8f81fece  South
Datacenter: North America
=====
Status=Up/Down
// State=Normal/Leaving/Joining/Moving
-- Address      Load          Tokens       Owns (effective)  Host ID                               Rack
UN 127.0.0.2     242.59 KiB    256          100.0%            cd34282f-e91c-493b-a74c-4ea11dfa24ff  US
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>
```

8.4. Nodetool ring:

It displays token ring information with the status of each of the nodes. The token ring is responsible for managing the partitioning of data within the Cassandra cluster. This command is critical if a cluster is facing data consistency issues. By default, `nodetool ring` displays all nodes that are involved in the ring and tokens that are assigned to each one of them. Optionally, you can specify the keyspace name and table name arguments to filter the output and display information.

Follow the below syntax to use this command:

```
nodetool ring [keyspace] [table]
```

```
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>nodetool ring
Datacenter: Asia
=====
Address      Rack      Status State   Load            Owns              Token
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          9208723366316796972
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -9181914919680777424
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -9135898857160291318
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -9124738933726578350
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -9106193222192747245
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -9090396655366360064
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -9036326451069183773
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8987406218220030025
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8868651564368241096
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8829507705743697056
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8711135424320361754
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8705362163446925636
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8618416343924162790
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8493121937336985691
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8442274297240864100
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8398844943476955648
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8358493682708979079
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8325850949216802965
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8311949547825747815
127.0.0.1     South     Up      Normal  241.51 KiB      100.00%          -8202389261199346109
```

8.5. Nodetool gossipinfo:

It provides the gossip protocol related statistics.

Follow the below syntax to use this command:

```
nodetool [options] gossipinfo
```

```
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>nodetool gossipinfo
/127.0.0.1
  generation:1735198167
  heartbeat:9476
  STATUS:25:NORMAL,-1199880975988446116
  LOAD:9460:196256.0
  SCHEMA:20:e84b6a60-24cf-30ca-9b58-452d92911703
  DC:9:Asia
  RACK:11:South
  RELEASE_VERSION:4:3.11.17
  INTERNAL_IP:7:127.0.0.1
  RPC_ADDRESS:3:127.0.0.1
  NET_VERSION:1:11
  HOST_ID:2:a0eadcc0-2bb4-4b09-8ae3-4b0e8f81fece
  RPC_READY:28:true
  SSTABLE_VERSIONS:5:big-me
  TOKENS:24:<hidden>
/127.0.0.2
  generation:1735198829
  heartbeat:8793
  STATUS:25:NORMAL,-104634129013279800
  LOAD:8777:196992.0
  SCHEMA:20:e84b6a60-24cf-30ca-9b58-452d92911703
  DC:9:North America
  RACK:11:US
  RELEASE_VERSION:4:3.11.17
  INTERNAL_IP:7:127.0.0.2
  RPC_ADDRESS:3:127.0.0.2
  NET_VERSION:1:11
  HOST_ID:2:cd34282f-e91c-493b-a74c-4ea11dfa24ff
  RPC_READY:36:true
  SSTABLE_VERSIONS:5:big-me
  TOKENS:24:<hidden>
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>
```

8.6. Nodetool getlogginglevels:

It gives logging levels defined in the database for all areas

Follow the below syntax to use this command:

```
nodetool [options] getlogginglevels
```

```
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>nodetool getlogginglevels

Logger Name                                Log Level
ROOT                                        INFO
com.thinkaurelius.thrift                  ERROR
org.apache.cassandra                      DEBUG
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>
```

8.7. Nodetool netstats:

This command provides the network information about the host machine.

Follow the below syntax to use this command:

```
nodetool [options] netstats
```

```
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>nodetool netstats
Mode: NORMAL
Not sending any streams.
Read Repair Statistics:
Attempted: 0
Mismatch (Blocking): 0
Mismatch (Background): 0
Pool Name           Active   Pending   Completed   Dropped
Large messages      n/a      0          0           0
Small messages      n/a      0         140         0
Gossip messages     n/a      0       29479       0
D:\ProgramFiles\Cassandra\apache-cassandra-3.11.17\bin>
```

8.8. Nodetool tablestats:

This command provides statistics about one or more tables. The table stats are updated when SSTables change through compaction or flushing. Cassandra uses the metrics-core library to make the output more informative and easier to understand.

Follow the below syntax to use this command:

```
nodetool [options] tablestats
```

8.9. Nodetool cleanup:

This command that is required to remove data from nodes when a new node is added to a cluster and token ring is re-distributed. After `cleanup` it is recommended to run `compact` command.

Follow the below syntax to use `cleanup` command:

```
nodetool [connection_options] cleanup [-j num_jobs] [--]
[keyspace_name table_name [table_name ...]]
```

8.10. Nodetool compact:

This command is useful to perform compaction to merge several SSTables (data files in Cassandra) and keep the resultant SSTable with the latest updated data.

When a data update or delete operation is triggered, Cassandra does not overwrite or delete the data. In case of updates, a different timestamped version of updates is maintained and in case of deletes, the data is marked for deletion as **tombstones** and then the latest version of

data is obtained post the merge of the SSTables. Therefore, it is important to perform compactions on a periodic basis in order to keep the cluster healthy.

Follow the below syntax to use `compact` command:

```
nodetool [options] compact [(-et <end_token> | --end-token
<end_token>)]
[(-s | --split-output)] [(-st <start_token> | --start-token
<start_token>)] [--] [<keyspace> [<tables>...]]
[--user-defined] <relative_path_to_SSTable file>
```

8.11. Nodetool decommission:

It decommissions the node where this command is executed, and the data of the node is streamed to the next node in the ring. It is one of the important commands that need to be executed when the cluster needs to be scaled to ensure no data loss.

Follow the below syntax to use `decommission` command:

```
nodetool [options] decommission
```

8.12. Nodetool drain:

It flushes all memtables (temporary tables usually on the heap) to the SSTables on disk. Once the command is executed, the node stops listening for connections from clients and other nodes. The node is marked as DS (Down/Stopped) in the cluster in the `nodetool status` command output. This command is usually run before any maintenance activities or when upgrading a node to a newer version of Cassandra.

Follow the below syntax to use `drain` command:

```
nodetool [options] drain
```

8.13. Nodetool garbagecollect:

It performs single SSTable compactions to eliminate updates or logically deleted data (Tombstones).

For each SSTable, the command creates a new SSTable with unnecessary data cleaned out. By default, `garbagecollect` removes rows or partitions that have been deleted or overwritten with newer data. It can also remove deleted or overwritten cell values when `-g` argument is specified. This command can eliminate expired tombstones which are older than `gc_grace_seconds`, but not fresh ones.

Note that `garbagecollect` with `-g` cell option incurs very high Disk I/O and hence it should be ensured that enough disk space is available.

Follow the below syntax to use this command:

```
nodetool options garbagecollect [--] keyspace_name table_name
```

8.14. Nodetool join:

It adds a new node to the cluster. When this command is executed, the new node will start to communicate with other nodes in the cluster and receive data from them. It is important to ensure that the new node has the same version of Cassandra as the existing nodes in the cluster.

Follow the below syntax to use this command:

```
nodetool <options> join
```

8.15. Nodetool removemode:

This command is useful when a node is no longer needed or when a node has failed and needs to be replaced or removed. Before removing the dead node, `nodetool decommission` command must be run first on the node being removed and then `nodetool removemode` command should be executed on the live node in the cluster.

When a node is removed, Cassandra redistributes the data that was stored on that node to the remaining nodes in the cluster. After the data has been redistributed, the node being removed will be marked as “removed” and will no longer participate in the cluster.

It is important to ensure that all data has been successfully redistributed before decommissioning or shutting down the node being removed. This can be checked using `nodetool status` and monitoring the “UN” (up and normal) status of all nodes in the cluster.

It is recommended to perform a full repair of the cluster after removing a node to ensure data consistency. This can be done using `nodetool repair` command.

Follow the below syntax to use `nodetool removemode` command:

```
nodetool [connection_options] removemode -- <status> | <force> | <ID>
```

8.16. Nodetool assassinate:

This command is used to assassinate a node and should be performed when the `nodetool removemode` command fails.

Follow the below syntax to use `nodetool assassinate` command:

```
nodetool [options] assassinate <ip_address>
```


8.17. Nodetool repair:

Since frequent data deletions and downtime in nodes may lead to data inconsistencies, `nodetool repair` ensures data consistency across all nodes in the cluster. It works by comparing the data on each node with the data on other nodes and resolving any inconsistencies. This is done by constructing a **merkle tree** whose leaves are the hashes of the individual keys.

Note that while `nodetool repair` is running, it can cause increased network traffic and disk I/O on the nodes in the cluster. This can impact the performance of other applications running on the same nodes. It can also cause temporary data unavailability during the repair process which can be mitigated by running repairs during off-peak hours or using incremental repair (new data since last repair) instead of full repair (complete data).

If `nodetool repair` is interrupted or fails for any reason, it can leave the cluster in an inconsistent state and so it is important to monitor the progress of `nodetool repair` and ensure that it completes successfully.

Follow the below syntax to use this command:

```
nodetool repair [-full|-inc]
```

Congratulations!! You have successfully installed Apache Cassandra with single node and multi-node configuration and executed database queries using CQLSH tool along with an overview of nodetool utility to monitor and manage Cassandra cluster in Windows operating system.