Apache Cassandra 3.0.9 Installation Guide

Unotech Software Pvt Ltd

Created by:	Guide by:
Nishith Padh Mahantesh. D	Dushyant. Min 3/29/2017

Index

Overview:

Installation Guide:

Overview:

Document Purpose:

This document is created with the purpose:

- Brief Introduction to the Cassandra
- Architecture
- Installation of the Cassandra-3.0.9 for distributed mode
- Configuring the Cassandra for distributed mode
- Introduction to cqlsh
- Basic tutorial for creating the Keyspace and Column-family
- Inserting the Data and writing basic cqlsh query

Brief Introduction to the Cassandra :

Apache Cassandra™ is a massively scalable NoSQL database. Cassandra's technical roots can be found at companies recognized for their ability to effectively manage big data – Google, Amazon, and Facebook – with Facebook open sourcing Cassandra to the Apache Foundation in 2009.

Apache Cassandra is

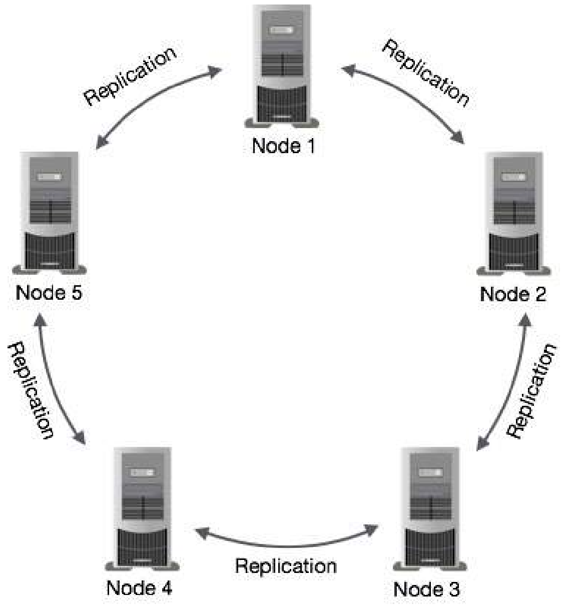
- Free
- Distributed
- High Performance
- Massively Scalable
- Fault Tolerant

The massive scale, high performance, and never-go-down nature of these applications has forged a new set of technologies that have replaced the legacy RDBMS, with O'Reilly describing the situation in this way:

"Big data is data that exceeds the processing capacity of conventional database systems. The data is too big, moves too fast, or doesn't fit the structures of your database architectures. To gain value from this data, you must choose an alternative way to process it."

Architecture:

- Cassandra provides automatic data distribution across all nodes that participate in a "ring" or database cluster.
- There is nothing programmatic that a developer or administrator needs to do or code to distribute data across a cluster
- Data is transparently partitioned across all nodes in either a randomized or ordered fashion, with random being the default.



- Cassandra was designed with the understanding that system/hardware failures can and do occur
- Rather than using a legacy master-slave or a manual and difficult-tomaintain sharded design, Cassandra has a peer-to-peer distributed architecture that is much more elegant, and easy to set up and maintain.
- In Cassandra, all nodes are the same; there is no concept of a master node, with all nodes communicating with each other via a gossip protocol.
- Cassandra's built-for-scale architecture means that it is capable of handling petabytes of information and thousands of concurrent users/operations per second (across multiple data centers) as easily as it can manage much smaller amounts of data and user traffic

Some of the application use cases that Cassandra excels in include:
• Real-time, big data workloads
• Time series data management
• High-velocity device data consumption and analysis
• Media streaming management (e.g., music, movies)
• Social media (i.e., unstructured data) input and analysis
• Online web retail (e.g., shopping carts, user transactions)
• Real-time data analytics • Online gaming (e.g., real-time messaging)
• Software as a Service (SaaS) applications that utilize web services
• Online portals (e.g., healthcare provider/patient interactions)
• Most write-intensive systems

Installation Guide:

Prerequisite:
Java 8 has to be installed before installing Cassandra and other basic utilities

Apache Cassandra 3.0.9 Installation:
Create the directory (Recommended more disk space) for simplicity we considering /opt

```
cd /opt
mkdir cassandra
cd cassandra
```

Download/copy the `apache-cassandra-3.9-bin.tar.gz`

```
wget  http://www-eu.apache.org/dist/cassandra/3.9/apache-cassandra-3.9-bin.tar.gz
```

Extract the tar

```
tar -zxvf apache-cassandra-3.9-bin.tar.gz
```

Setting the environment variables in bashrc

```
vim ~/.bashrc
#Cassandra Environment Variables Starts
export CASSANDRA_HOME=/opt/cassandra/apache-cassandra-3.9
export PATH=$CASSANDRA_HOME/bin:$PATH
#Cassandra Environment Variables ends
source ~/.bashrc
```

Create the folders - data, commitlog, saved_caches

```
cd /opt/cassandra
mkdir data
mkdir commitlog
mkdir saved_caches
```

Configuration:

Configuring the Cassandra for distributed mode

```
cd $CASSANDRA_HOME/conf/
vim cassandra.yaml
```

Set the directories created

search for 'data_file_directories'

```
/data_file_directories
data_file_directories: /opt/cassandra/data
```

search for 'commitlog_directory'

```
/commitlog_directory
commitlog_directory: /opt/cassandra/commitlog
```

search for 'saved_caches_directory'

```
/saved_caches_directory
saved_caches_directory: /opt/cassandra/saved_caches
```

Search seeds

```
/seeds
-    seeds:    "<ip-of-Cassandranode1>,<ip-of-Cassandranode2>,…,<ip-of-Cassandranoden>"
```

Note: Make any 3 cassandra nodes as seeds. Mention ips of those nodes in the seeds
Example: if have Cassandra installed from x.x.x.10-x.x.x.20 and your Cassandra seeds are x.x.x.11, x.x.x.15, x.x.x.18, then its look like this –

- seeds: “x.x.x.11, x.x.x.15, x.x.x.18”

Seeds will remain same in all the nodes

Search for listen_address

```
/listen_address
listen_address: <ip of the local machine>
```

Search for rpc_address

```
/rpc_address
```

```
rpc_address: <ip of the local machine>
```

Search for rpc_start

```
/rpc_start
rpc_start: true
```

Follow the same instruction for installing Cassandra in all the nodes

Starting the Cassandra:

Once installation is done in all the nodes,
First start the Cassandra in seed nodes followed by other nodes,

```
cd $CASSANDRA_HOME
./bin/cassandra
```

After starting the Cassandra in all the nodes check the status

```
cd $CASSANDRA_HOME
./bin/nodetoolstatus
```

It will look similar to this

```
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
-- Address Load Tokens Owns (effective) Host ID Rack
UN 172.17.0.3 285.2 KiB 256 70.1% f1a6e431-03cd-4baa-8799-206715cfd8f6 rack1
UN 172.17.0.2 268.57 KiB 256 65.8% f4fded89-f538-4f19-baaa-63b0067cf7ab rack1
UN 172.17.0.4 362.14 KiB 256 64.1% 42f30a7e-65c5-469e-9549-459cbd6bc910 rack1
```

Cqlsh:

Once the Cassandra is started in all nodes, lets create the keyspace(Database in RDBMS) and column-family(table in RDBMS) in cassandra

Start the cqlsh

```
cqlsh <ip of the Cassandra node>
```

Creating Keyspace and Column family:

Creating Keyspace

```
cqlsh>CREATE KEYSPACE test WITH replication = {'class': SimpleStrategy', 'replication_factor' : 1};
```

Note: replication factor depends on number of Cassandra nodes. Replication factor is always less than or equal number of Cassandra nodes

```
cqlsh>USE test;
```

Creating the Column-family

```
cqlsh:test>CREATE TABLE test1(emp_id int PRIMARY KEY, emp_name text, emp_city text);
```

Insert the data

```
cqlsh:test>INSERT INTO test1 (emp_id, emp_city, emp_name) VALUES ( 1,
'Rohan', 'Mumbai');
```

Do SELECT

```
cqlsh:test> SELECT * FROM test1;
```

Output:

```
emp_id | dmp_name | emp_city
--------+-----------+----------
 1 | Mumbai | Rohan
```

Further References:

https://academy.datastax.com/
https://www.tutorialspoint.com/cassandra/
https://www.datastax.com/resources/tutorials

Courtesy:

Courtesy for Brief introduction for Cassandra and Architecture

https://www.datastax.com/wp-content/uploads/2012/08/WP-IntrotoCassandra.pdf
https://www.slideshare.net/DataStax/an-overview-of-apache-cassandra

Image Courtesy

https://www.tutorialspoint.com/cassandra/cassandra_architecture.htm