### Installation from Debian packages

* For the <release series> specify the major version number, without dot, and with an appended x.
* The latest <release series> is 40x.
* For older releases, the <release series> can be one of 311, 30x, or 22x.
* Add the Apache repository of Cassandra to /etc/apt/sources.list.d/cassandra.sources.list, for example for the latest 4.0

echo "deb https://debian.cassandra.apache.org 40x main" | sudo tee -a /etc/apt/sources.list.d/cassandra.sources.list

* Add the Apache Cassandra repository keys:

curl https://downloads.apache.org/cassandra/KEYS | sudo apt-key add -

* Update the repositories:

sudo apt-get update

* If you encounter this error:

GPG error: http://www.apache.org 311x InRelease: The following signatures couldn't be verified because the public key is not available: NO\_PUBKEY A278B781FE4B2BDA

Then add the public key A278B781FE4B2BDA as follows:

sudo apt-key adv --keyserver pool.sks-keyservers.net --recv-key A278B781FE4B2BDA

and repeat sudo apt-get update. The actual key may be different, you get it from the error message itself. For a full list of Apache contributors public keys, you can refer to [Cassandra KEYS](https://downloads.apache.org/cassandra/KEYS).

* Install Cassandra:

sudo apt-get install cassandra

* You can start Cassandra with sudo service cassandra start and stop it with sudo service cassandra stop. However, normally the service will start automatically. For this reason be sure to stop it if you need to make any configuration changes.
* Verify that Cassandra is running by invoking nodetool status from the command line.
* The default location of configuration files is /etc/cassandra.
* The default location of log and data directories is /var/log/cassandra/ and /var/lib/cassandra.
* Start-up options (heap size, etc) can be configured in /etc/default/cassandra.

### Installation from RPM packages

* For the <release series>`` specify the major version number, without dot, and with an appended x.
* The latest <release series> is 40x.
* For older releases, the <release series> can be one of 311x, 30x, or 22x.
* (Not all versions of Apache Cassandra are available, since building RPMs is a recent addition to the project.)
* For CentOS 7 and similar (rpm < 4.14), append the noboolean repository
* Add the Apache repository of Cassandra to /etc/yum.repos.d/cassandra.repo, for example for the latest 4.0 version:

[cassandra]

name=Apache Cassandra

baseurl=https://redhat.cassandra.apache.org/40x/

gpgcheck=1

repo\_gpgcheck=1

gpgkey=https://downloads.apache.org/cassandra/KEYS

Or for CentOS 7:

[cassandra]

name=Apache Cassandra

baseurl=https://redhat.cassandra.apache.org/40x/noboolean/

gpgcheck=1

repo\_gpgcheck=1

gpgkey=https://downloads.apache.org/cassandra/KEYS

* Install Cassandra, accepting the gpg key import prompts:

sudo yum install cassandra

Start Cassandra (will not start automatically):

service cassandra start

Systemd based distributions may require to run systemctl daemon-reload once to make Cassandra available as a systemd service. This should happen automatically by running the command above.

Make Cassandra start automatically after reboot:

chkconfig cassandra on

Please note that official RPMs for Apache Cassandra only have been available recently and are not tested thoroughly on all platforms yet. We appreciate your feedback and support and ask you to post details on any issues in the corresponding Jira ticket.

## Source

Development is done in the Apache Git repository. To check out a copy:

git clone https://gitbox.apache.org/repos/asf/cassandra.git

—-------------------------------------------------------------------------------------------------------------------------

**Installation cassandra as tar file :-**

<https://dlcdn.apache.org/cassandra/4.0.7/apache-cassandra-4.0.7-bin.tar.gz>  
—----------------------------------------------------------------------------------------------------------------------------

### STEP 1: GET CASSANDRA USING DOCKER

docker pull cassandra:latest

### STEP 2: START CASSANDRA

A Docker network allows us to access the container’s ports without exposing them on the host.

docker network create cassandra

docker run --rm -d --name cassandra --hostname cassandra --network cassandra cassandra

### STEP 3: CREATE FILES

The Cassandra Query Language (CQL) is very similar to SQL but suited for the JOINless structure of Cassandra.

Create a file named data.cql and paste the following CQL script in it. This script will create a keyspace, the layer at which Cassandra replicates its data, a table to hold the data, and insert some data into that table:

-- Create a keyspace

CREATE KEYSPACE IF NOT EXISTS store WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication\_factor' : '1' };

-- Create a table

CREATE TABLE IF NOT EXISTS store.shopping\_cart (

userid text PRIMARY KEY,

item\_count int,

last\_update\_timestamp timestamp

);

-- Insert some data

INSERT INTO store.shopping\_cart

(userid, item\_count, last\_update\_timestamp)

VALUES ('9876', 2, toTimeStamp(now()));

INSERT INTO store.shopping\_cart

(userid, item\_count, last\_update\_timestamp)

VALUES ('1234', 5, toTimeStamp(now()));

### STEP 4: LOAD DATA WITH CQLSH

The CQL shell, or cqlsh, is one tool to use in interacting with the database. We’ll use it to load some data into the database using the script you just saved.

docker run --rm --network cassandra -v "$(pwd)/data.cql:/scripts/data.cql" -e CQLSH\_HOST=cassandra -e CQLSH\_PORT=9042 -e CQLVERSION=3.4.5 nuvo/docker-cqlsh

Note: The cassandra server itself (the first docker run command you ran) takes a few seconds to start up. The above command will throw an error if the server hasn’t finished its init sequence yet, so give it a few seconds to spin up.

### STEP 5: INTERACTIVE CQLSH

Much like an SQL shell, you can also of course use CQLSH to run CQL commands interactively.

docker run --rm -it --network cassandra nuvo/docker-cqlsh cqlsh cassandra 9042 --cqlversion='3.4.5'

This should get you a prompt like so:

Connected to Test Cluster at cassandra:9042.

[cqlsh 5.0.1 | Cassandra 4.0.4 | CQL spec 3.4.5 | Native protocol v5]

Use HELP for help.

cqlsh>

### STEP 6: READ SOME DATA

SELECT \* FROM store.shopping\_cart;

### STEP 7: WRITE SOME MORE DATA

INSERT INTO store.shopping\_cart (userid, item\_count) VALUES ('4567', 20);

### STEP 8: CLEAN UP

docker kill cassandra

docker network rm cassandra

—---------------------------------------------------------------

## What is Apache Cassandra?

### Apache Cassandra is an open source NoSQL distributed database trusted by thousands of companies for scalability and high availability without compromising performance. Linear scalability and proven fault-tolerance on commodity hardware or cloud infrastructure make it the perfect platform for mission-critical data.

### **Hybrid**

Masterless architecture and low latency means Cassandra will withstand an entire data center outage with no data loss—across public or private clouds and on-premises.

### **Fault Tolerant**

Cassandra’s support for replicating across multiple datacenters is best-in-class, providing lower latency for your users and the peace of mind of knowing that you can survive regional outages. Failed nodes can be replaced with no downtime.

### **Performant**

Cassandra [**consistently outperforms**](http://vldb.org/pvldb/vol5/p1724_tilmannrabl_vldb2012.pdf) popular NoSQL alternatives in benchmarks and real applications, primarily because of fundamental architectural choices.

### **Distributed**

Cassandra is suitable for applications that can’t afford to lose data, even when an entire data center goes down. There are no single points of failure. There are no network bottlenecks. Every node in the cluster is identical.

### **Scalable**

Read and write throughput both increase linearly as new machines are added, with no downtime or interruption to applications.

### **Elastic**

Cassandra streams data between nodes during scaling operations such as adding a new node or datacenter during peak traffic times. Zero Copy Streaming makes this up to 5x faster without vnodes for a more elastic architecture particularly in cloud and Kubernetes environments.

## What is Apache Cassandra?

Cassandra is a NoSQL distributed database. By design, NoSQL databases are lightweight, open-source, non-relational, and largely distributed. Counted among their strengths are horizontal scalability, distributed architectures, and a flexible approach to schema definition.

NoSQL databases enable rapid, ad-hoc organization and analysis of extremely high-volume, disparate data types. That’s become more important in recent years, with the advent of Big Data and the need to rapidly scale databases in the cloud. Cassandra is among the NoSQL databases that have addressed the constraints of previous data management technologies, such as SQL databases.

### Distribution provides power and resilience

Since it is a distributed database, Cassandra can (and usually does) have multiple nodes. A node represents a single instance of Cassandra. These nodes communicate with one another through a protocol called gossip, which is a process of computer peer-to-peer communication. Cassandra also has a masterless architecture – any node in the database can provide the exact same functionality as any other node – contributing to Cassandra’s robustness and resilience. Multiple nodes can be organized logically into a cluster, or "ring". You can also have multiple datacenters.

One reason for Cassandra’s popularity is that it enables developers to scale their databases dynamically, using off-the-shelf hardware, with no downtime. You can expand when you need to – and also shrink, if the application requirements suggest that path.

Cassandra makes it easy to increase the amount of data it can manage. Because it’s based on nodes, Cassandra scales horizontally (aka scale-out), using lower commodity hardware.

### Replication ensures reliability and fault tolerance

Replication ensures that data isn’t lost. If a request comes in for data, even if one of our replicas has gone down, the other two are still available to fulfill the request. The coordinator stores a “hint” for that data as well, and when the downed replica comes back up, it will find out what it missed, and catch up to speed with the other two replicas. No manual action is required, this is done completely automatically.

### Tuning your consistency