|  |
| --- |
|  |
| ***Apache Cassandra 3.11.1*** |
|  |

|  |
| --- |
| **Delsey Sabu** |

Table of Contents

[Product Background 2](#_Toc499896995)

[Product Name and Version 2](#_Toc499896996)

[Company Name 2](#_Toc499896997)

[Product background and product history/highlights 2](#_Toc499896998)

[Who uses the product 2](#_Toc499896999)

[Data model supported 2](#_Toc499897000)

[System Requirements 2](#_Toc499897001)

[DBMS Functions 4](#_Toc499897002)

[Data storage, retrieval and update 4](#_Toc499897003)

[Integrity services 4](#_Toc499897004)

[User-accessible catalog 4](#_Toc499897005)

[Services to promote data independence 4](#_Toc499897006)

[Database tuning and performance 4](#_Toc499897007)

[Authorization services 5](#_Toc499897008)

[Transaction support 5](#_Toc499897009)

[Concurrency control services 5](#_Toc499897010)

[Recovery services 5](#_Toc499897011)

[Support for data communication/ networks/ internet 6](#_Toc499897012)

[Utility Services 6](#_Toc499897013)

[Evaluation 8](#_Toc499897014)

[Bibliography 10](#_Toc499897015)

# Product Background

# Product Name and Version

Apache Cassandra, version 3.11.1

# Company Name

Apache Software Foundation

# Product background and product history/highlights

Apache Cassandra was originally started as a project at Facebook to power their Inbox Search feature. It was originally built using Amazon’s Dynamo and Google’s BigTable. Cassandra was first released as an open source project in July 2008, in 2009 became an Apache incubator project, and in 2010 it became an Apache top-level project. Since becoming a top-level project, Cassandra has had 11 major releases with the most current being 3.11. The purpose of Cassandra is to manage large amount of data across many commodity servers while also delivering high availability and no single point of failure. There is a theorem known as The CAP Theorem which states that it is impossible for a distributed computer system to provide Consistency, Availability, and Partitioning all at the same time. Cassandra is classified as an AP system but can be configure to be a CAP system, meaning that it can provide all three guarantees. Furthermore, Cassandra is known as a “masterless” service meaning that all of its nodes are the same. This allows for automatic data distribution across all nodes in its database cluster. Cassandra makes it simple for replication to occur through this feature.

# Who uses the product

There are many large well-known companies that rely on Apache Cassandra to maintain their information. Apple, Comcast, eBay, Instagram, Spotify, Uber, Netflix, Hulu, Reddit, GitHub, Intuit, and The Weather Channel are just a few that use Cassandra to store large amounts of information. Cassandra is one of the best solutions for companies that need to handle large amounts of data and operations all at the same time.

# Data model supported

Wide column store

# System Requirements

* + Minimal production servers require at least 2 cores and 8GB of RAM.
  + Specific hardware is not necessary. The only requirement is that it has access to at least 2 cores and 8GB of RAM.
  + BSD, Linux, OS X, Windows
* DB size supported
* Apache Cassandra is an open source free software. There are third party applications that offer different cluster hosting services.
* Cassandra can manage large amounts of data across multiple servers while maintaining high availability. Being a masterless service, Cassandra can replicate information constantly for quick and easy recovery if anything bad were to occur. Furthermore, Cassandra is comparatively fast allowing for fast linear scale performance, moreover it allows for fast response time that customers expect.

# DBMS Functions

## Data storage, retrieval and update

Basic Cassandra DDLs include CREATE KEYSPACE, ALTER KEYSPACE, DROP KEYSPACE, and CREATE TABLE. Some basic DMLs include SELECT, UPDATE, INSERT, DELETE, and BATCH. Apache Cassandra is a “language oriented” program that uses the Cassandra Query Language or CQL, it’s very similar to SQL. There are quite a few of host languages that Cassandra supports, the most popular languages include, C#, C++, Java, JavaScript, Perl, PHP, Ruby, and Python. Cassandra does allow the creation of user defined functions. These functions must be created prior to when it is first used in a SELECT statement. In order to use user-defined functions with Java or Javascript, just set *enable\_user\_defined\_function* to *true* in the Cassandra.yaml file. Other scripting languages such as Python, Ruby, and Scala can be used as well, only it requires adding a JAR to the classpath.

## Integrity services

Cassandra supports Primary Keys, Partition Keys, and Clustering Keys. Just like a relational database, a Primary Key in Cassandra must be unique and must contain a value. In Cassandra they can be either simple or compound. A Compound Primary Key contains multiple columns, while a Simple Primary Key contains only one column. Cassandra also uses the first column of the Primary Key as the Partition Key. This Partition Key which node in the cluster will store that row. If the Primary Key is simple, the Primary Key serves also as the Partition Key. Furthermore, Cassandra allows the use of Triggers. These can be defined with a name and given a statement to execute.

## User-accessible catalog

There is a lot of information in the Cassandra system catalog. Mainly it consists of information on how to operate their service. Information such as how to set up replication between servers, how to use their CQL queries and other functions. All of this information is available to the user through the Apache Cassandra website in their documentation. As Cassandra 3.11.1 is very new, there is new data documentation being released still.

## Services to promote data independence

Apache Cassandra, Hadoop, and Solr can be all be configure to across multiple data centers and cloud services. This allows for the fastest response times to both customers and employees needing access. This also allows for an extra level of security. In the case that one data center goes down, there are others that can pick up the slack.

## Database tuning and performance

A very popular tool for testing with Cassandra is cstar\_perf that was built by Datastax. This tool provides users with helpful insight into the performance of their Cassandra databases. Cassandra also offers the *Cassandra-stress* tool. It a Java-based stress testing tool for basic benchmarking and load testing a Cassandra cluster. There are also a few monitoring tools for Cassandra such as RazorSQL that boasts many features such Export/Import tools, backup tools, and a table editor that automatically generates an appropriate INSERT, UPDATE, or DELETE statement.

## Authorization services

Within Cassandra Role Based Access control (RBAC), the concept of “ROLE” replaces that of “USER,” where a “ROLE” refers to single user or a group of users that have the same access permissions. A role can be created, altered, dropped and listed through CQL statements. Cassandra only supports internal authentication features. The default authorization in Cassandra grants all permissions to all roles – “AllowAllAuthenticator”, which does not enforce authentication at all. Cassandra also has full permissions management functionality and stores the data in system tables. The default assumption for permissions is that a given role has no access to any database resources, meaning that all requests will be rejected until the needed/required permissions have been allowed to the role.

## Transaction support

Cassandra does not use ACID transactions with rollback or locking mechanisms. Cassandra uses atomic, isolated, and durable transactions with consistency that lets the users decide the strength in which the want each transaction to be. It is important to note that Cassandra does not support joins or foreign keys and thus does not give consistency in the case of ACID. The write operations in Casandra are atomic at the partition level, hence, insertions or updates in two or more rows in the same partition is counted as one write operation. Casandra also has lightweight or compare and set transactions, which are used in situations that require nodes in the distributed system to agree on changes in data.

## Concurrency control services

As said before, Cassandra does not use ACID transactions with rollback or locking mechanisms. Client side time stamps are used to figure out the most recent update to a column. The most current or latest timestamp always wins when requesting data, so that the most recent update is the one that will eventually keep. Cassandra does not support transactions such like in the case of bundling multiple row updates into one all-or-nothing operation. It also does not roll back when a write succeeds on one replica, but fails on other replicas. Writes in Cassandra are also durable, meaning that all writes to a replica node are recorded in memory and on disk before the writes are acknowledged as successful. If a failure occurs in the system the commit log is replayed to recover any lost writes. Write and delete operations in Cassandra are performed with full row-level isolation, meaning that a write to a row within a single partition on a single node is only seen by clients performing the operation and it is restricted like that until the operation is complete.

## Recovery services

Cassandra is a peer-to-peer, fault-tolerant system, where data is replicated among multiple nodes across multiple data centers, thus, single or more node failures can be recovered from surviving nodes with the data. As long as there is at least one node with the replicated data, Cassandra can recover the data without needing an external source. It is still necessary to do backups to recover any errors made in data edits by client applications. A point-in-time recovery is done in the event of data corruption or some other disastrous situation. Cassandra keeps data in SSTable files and these are written to when Cassandra fills its memtable. New SSTable files are in key space directories as its memTable is filled and emptied. While the memTable is written to an SSTable file no more writes are made into that file. A “point-in-time” recovery requires recovery of all the SSTable files in a key space exactly as they were in a given instant. Cassandra has a snapshot utility that flushes all-in memory writes to a disk, then hard link each current SSTable file for each key space in snapshots sub directory in the local disk key space area. It also have incremental backup features, which reduces disk space requirements because Cassandra adds new files and deletes old ones as needed. Restoring Cassandra key space means restoring all the SSTable files as they existed, and although Cassandra does not provide a native restore utility, it does provide a restore procedure.

(Procedure taken from techblog)

*For each node in the cluster:*

1. *Shut down Cassandra.*
2. *Clear all files in commitlog directory (path defined by the <CommitLogDirectory> parameter in the cassandra.yaml file, by default /var/lib/cassandra/commitlog). Ideally, logs will be flushed before Cassandra is shut down, as the commitlog directory is a shared resource of all keyspaces, not just the one to be restored.*
3. *Removing all current contents of the active keyspace (all \*.db files).*
4. *Copying contents of desired snapshot to active keyspace.*
5. *Only if restored snapshot is the latest one, and you want the latest backup, copy contents of backup directory into active keyspace area on top of the restored snapshot files.*

Note that the process must be executed on all nodes in the cluster, otherwise nodes that did not get the restored data will “update” the restored nodes with the newer, bad data.

There are also multiple extensions to Cassandra native backup tools; automation, non-local storage, and multi-day/instance retentions. These are achieved through using bash, Puppet, Cron, and NFS. Cassandra does need a lot of space for backups.

## Support for data communication/ networks/ internet

It is possible for client applications to communicate with Cassandra via a public or private network. An application that tries to connect to Cassandra is given a list of initial endpoints and the application will try to establish an initial connection to each of the endpoints given, until a successful connection is made or all endpoints were tried. When the driver successfully establishes a connection to one of the nodes, the node will send back the list of IPs of all the other nodes in the cluster. As a result, when executing queries, the driver will use this list of IP, as well as the IP used to initiate the connection (one of the IP specified in “addContactPoints”). The list of IP address returned by the node is constructed using the cassandra.yaml parameter “**broadcast\_rpc\_address”** configured on each node of the cluster. “**broadcast\_rpc\_address” can be set to public or private IP. There is no way to** configure it to conditionally broadcast the list of public IP when a connection is established via the public interface, and broadcast the list of private IP when a connection is established via the private interface. Datastax Cassandra driver provides an interface called AddressTranslater which is meant to translate all InetScoketAddresses corresponding to the IP address found during node discovery, thus, making it easier to build a map that will translate private IP address to the corresponding public one.

## Utility Services

Cassandra start-up parameters can be run from the command line (Tarball installations) or specified in the [cassandra-env.sh](https://docs.datastax.com/en/cassandra/2.1/cassandra/tools/toolsCUtility_t.html#toolsCUtility_t__cassandraenvsh) file. Three utility services examples are;

**Clear gossip state when starting a node:**

* Command line: bin/cassandra -Dcassandra.load\_ring\_state=false
* cassandra-env.sh: JVM\_OPTS="$JVM\_OPTS -Dcassandra.load\_ring\_state=false"

**Start Cassandra on a node and do not join the cluster:**

* Command line: bin/cassandra -Dcassandra.join\_ring=false
* cassandra-env.sh: JVM\_OPTS="$JVM\_OPTS -Dcassandra.join\_ring=false"

**Replacing a dead node:**

* Command line: bin/cassandra -Dcassandra.replace\_address=10.91.176.160
* cassandra-env.sh: JVM\_OPTS="$JVM\_OPTS -Dcassandra.replace\_address=10.91.176.160"

# Evaluation

This table shows a quick comparison of RDBMS and a NoSQL database Cassandra.

| **Relational Database** | **Cassandra** |
| --- | --- |
| Handles moderate incoming data velocity | Handles high incoming data velocity |
| Data arriving from one/few locations | Data arriving from many locations |
| Manages primarily structured data | Manages all types of data |
| Supports complex/nested transactions | Supports simple transactions |
| Single points of failure with failover | No single points of failure; constant uptime |
| Supports moderate data volumes | Supports very high data volumes |
| Centralized deployments | Decentralized deployments |
| Data written in mostly one location | Data written in many locations |
| Supports read scalability (with consistency sacrifices) | Supports read and write scalability |
| Deployed in vertical scale up fashion | Deployed in horizontal scale out fashion |

Cassandra’s hierarchy of caching mechanisms and carefully done disk I/O ensures speed and data safety. Cassandra has massively scalable architecture with design where all nodes are the same, making it simple. Cassandra enables all nodes to be written to and read from. There is an increase in performance because of the ability to add nodes without going down. The continuous ability offers redundancy of both data and node function, which takes out single points of failure and provide constant uptime. Nodes that fail can be easily restored or replaced because of transparent fault detection and recovery. There is strong data protection though its commit log design that makes sure that there are no data loss and built in security. And above all, there is data compression without it affecting performance overhead. Cassandra excels at some use cases including; internet of things applications, product catalogs and retail apps, user activity tracking and monitoring, messaging, social media analytics and recommendation settings.

So far we talked about the pros of Cassandra, but Cassandra has it weaknesses as well. Cassandra employs “No Ad-Hoc Queries” meaning that one must model the data around queries one wants to surface, rather than around the structure of the data itself. There is also concerns with unpredictable performance because Cassandra has many different asynchronous jobs and background tasks that are not scheduled by the user, the performance can be unpredictable. This means that you may see performance impacts that may not be related to a query, or volume of queries.  This can make troubleshooting performance issues rather difficult. Newer versions of Cassandra will have limited support for aggregations with a single partition.  This is of very limited use. Thus, if doing ad-hoc analysis is a requirement for your application then Cassandra may not be for you.

# Bibliography

* “Cassandra and Backups.” *Constant Contact Tech Blog*, 18 Feb. 2013, techblog.constantcontact.com/devops/cassandra-and-backups/.
* *Cassandra System Properties*, db-engines.com/en/system/cassandra.
* “CassandraHardware.” *CassandraHardware - Cassandra Wiki*, wiki.apache.org/cassandra/CassandraHardware.
* “Hardware Choices¶.” *Documentation*, cassandra.apache.org/doc/latest/operating/hardware.html.
* “Introduction to Multi-Data Center Operations with Apache Cassandra, Hadoop, and Solr.” *DataStax: Always-on Data Platform | NoSQL | Apache Cassandra*, www.datastax.com/resources/whitepapers/intro-to-multidc.
* “Learn Cassandra.” *Where Is Data Stored | Learn Cassandra*, teddyma.gitbooks.io/learncassandra/content/model/where\_is\_data\_stored.html.
* Pandaforme. “Introduction To Cassandra.” *Understand the Cassandra Data Model |* Introduction To Cassandra, pandaforme.gitbooks.io/introduction-to-cassandra/content/understand\_the\_cassandra\_data\_model.html.
* “What Is Apache Cassandra™?” *What Is Apache Cassandra | DataStax Academy: Free Cassandra Tutorials and Training*, academy.datastax.com/planet-cassandra/what-is-apache-cassandra.