Cassandra Basics

Tushar B. Kute,
http://tusharkute.com

What is Cassandra?

- Apache Cassandra is highly scalable, high performance, distributed NoSQL database.
- Cassandra is designed to handle huge amount of data across many commodity servers, providing high availability without a single point of failure.
- Cassandra has a distributed architecture which is capable to handle a huge amount of data.
- Data is placed on different machines with more than one replication factor to attain a high availability without a single point of failure.

What is Cassandra?

- Cassandra is a column-oriented database.
- Cassandra is scalable, consistent, and fault-tolerant.
- Cassandra's distribution design is based on Amazon's Dynamo and its data model on Google's Bigtable.
- Cassandra is created at Facebook. It is totally different from relational database management systems.
- Cassandra follows a Dynamo-style replication model with no single point of failure, but adds a more powerful "column family" data model.
- Cassandra is being used by some of the biggest companies like Facebook, Twitter, Cisco, Rackspace, ebay, Twitter, Netflix, and more.

Cassandra

- Cassandra was initially developed at Facebook by two Indians Avinash Lakshman (one of the authors of Amazon's Dynamo) and Prashant Malik. It was developed to power the Facebook inbox search feature.
- The following points specify the most important happenings in Cassandra history:
  - Cassandra was developed at Facebook by Avinash Lakshman and Prashant Malik.
  - It was developed for Facebook inbox search feature.
  - It was open sourced by Facebook in July 2008.
  - It was accepted by Apache Incubator in March 2009.
  - Cassandra is a top level project of Apache since February 2010.
  - The latest version of Apache Cassandra is 3.2.1.

Cassandra: Features

- High Scalability
  - Cassandra is highly scalable which facilitates you to add more hardware to attach more customers and more data as per requirement.
- Rigid Architecture
  - Cassandra has not a single point of failure and it is continuously available for business-critical applications that cannot afford a failure.

Cassandra: Features

- Fast Linear-scale Performance
  - Cassandra is linearly scalable. It increases your throughput because it facilitates you to increase the number of nodes in the cluster. Therefore it maintains a quick response time.
- Fault tolerant
  - Cassandra is fault tolerant. Suppose, there are 4 nodes in a cluster, here each node has a copy of same data. If one node is no longer serving then other three nodes can served as per request.

Cassandra: Features

- Flexible Data Storage
  - Cassandra supports all possible data formats like structured, semi-structured, and unstructured. It facilitates you to make changes to your data structures according to your need.
- Easy Data Distribution
  - Data distribution in Cassandra is very easy because it provides the flexibility to distribute data where you need by replicating data across multiple data centers.

Cassandra: Features

- Transaction Support
  - Cassandra supports properties like Atomicity, Consistency, Isolation, and Durability (ACID).
- Fast writes
  - Cassandra was designed to run on cheap commodity hardware. It performs blazingly fast writes and can store hundreds of terabytes of data, without sacrificing the read efficiency.

Cassandra: Architecture

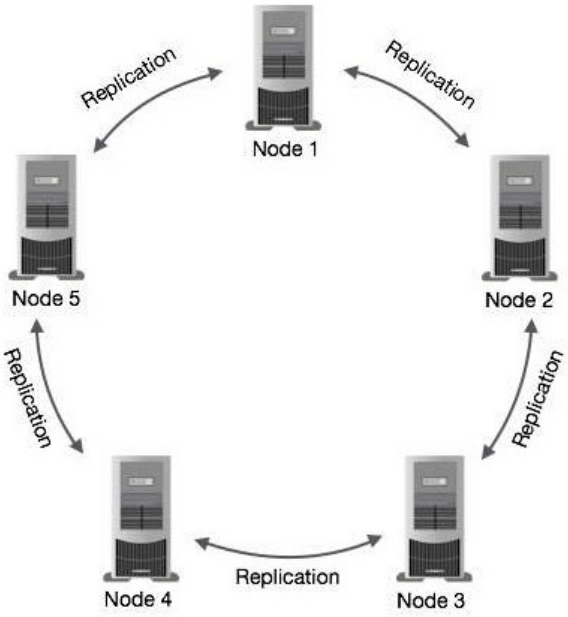
- Cassandra was designed to handle big data workloads across multiple nodes without a single point of failure. It has a peer-to-peer distributed system across its nodes, and data is distributed among all the nodes in a cluster.
  - In Cassandra, each node is independent and at the same time interconnected to other nodes. All the nodes in a cluster play the same role.
  - Every node in a cluster can accept read and write requests, regardless of where the data is actually located in the cluster.
  - In the case of failure of one node, Read/Write requests can be served from other nodes in the network.

Cassandra: Replication

- In Cassandra, nodes in a cluster act as replicas for a given piece of data.
- If some of the nodes are responded with an out-of-date value, Cassandra will return the most recent value to the client.
- After returning the most recent value, Cassandra performs a read repair in the background to update the stale values.

Cassandra: Replication

- See the following image to understand the schematic view of how Cassandra uses data replication among the nodes in a cluster to ensure no single point of failure.



Cassandra: Components

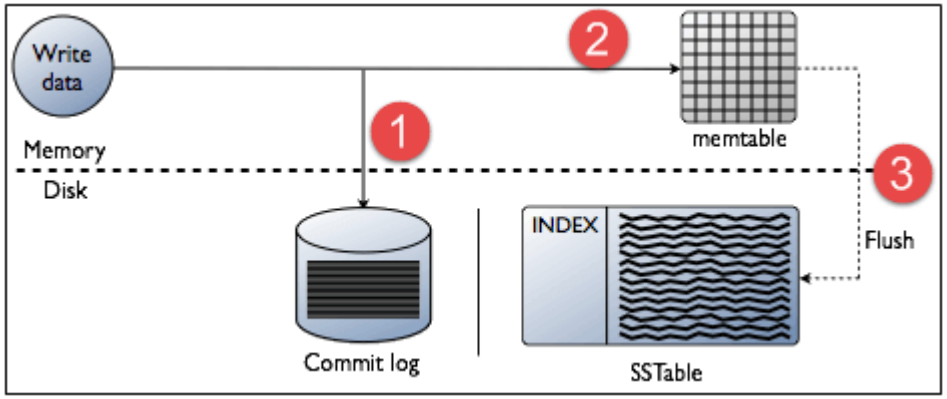
- Node: A Cassandra node is a place where data is stored.
- Data center: Data center is a collection of related nodes.
- Cluster: A cluster is a component which contains one or more data centers.
- Commit log: In Cassandra, the commit log is a crash-recovery mechanism. Every write operation is written to the commit log.
- Mem-table: A mem-table is a memory-resident data structure. After commit log, the data will be written to the mem-table. Sometimes, for a single-column family, there will be multiple mem-tables.
- SSTable: It is a disk file to which the data is flushed from the mem-table when its contents reach a threshold value.
- Bloom filter: These are nothing but quick, nondeterministic, algorithms for testing whether an element is a member of a set. It is a special kind of cache. Bloom filters are accessed after every query.

Cassandra Query Language CQL

- Cassandra Query Language (CQL) is used to access Cassandra through its nodes.
- CQL treats the database (Keyspace) as a container of tables.
- Programmers use cqlsh: a prompt to work with CQL or separate application language drivers.
- The client can approach any of the nodes for their read-write operations.
- That node (coordinator) plays a proxy between the client and the nodes holding the data.

Write Operation

- Every write activity of nodes is captured by the commit logs written in the nodes. Later the data will be captured and stored in the mem-table.
- Whenever the mem-table is full, data will be written into the SStable data file. All writes are automatically partitioned and replicated throughout the cluster. Cassandra periodically consolidates the SSTables, discarding unnecessary data.



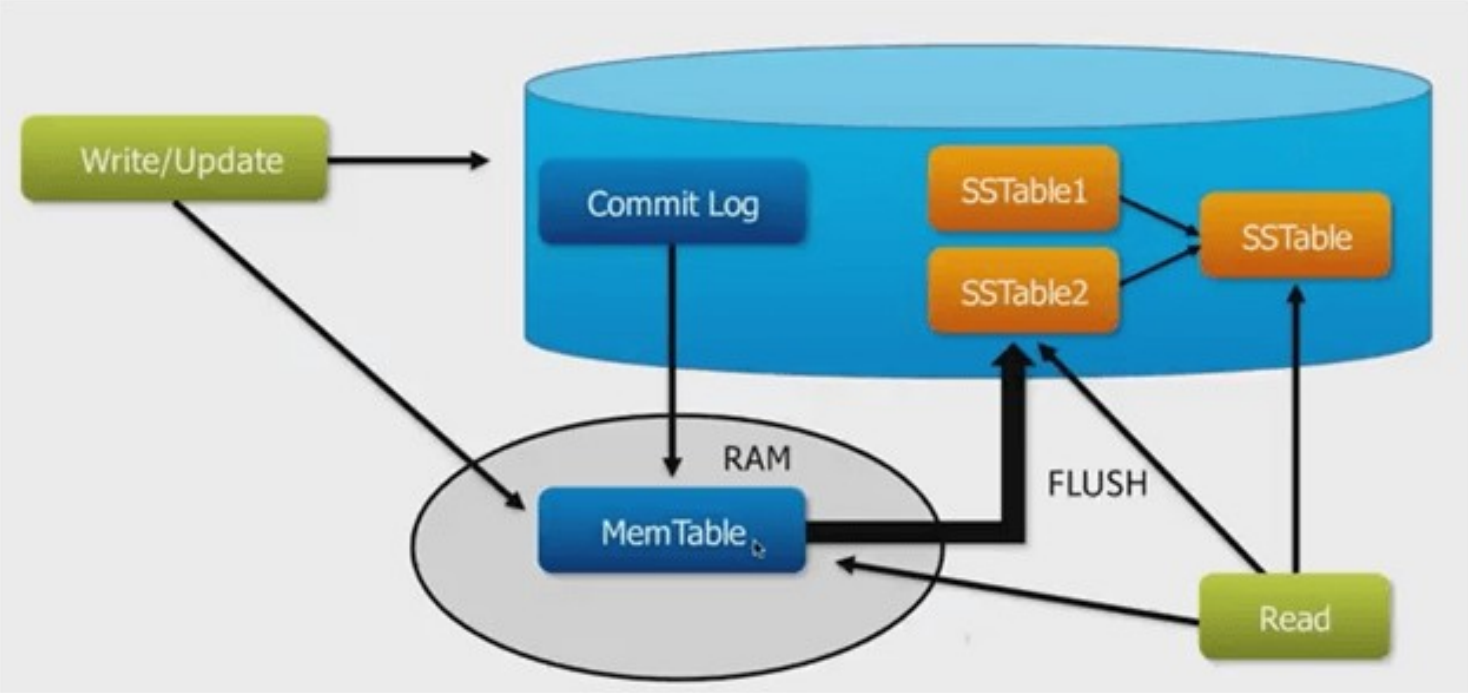
Read Operation

- In Read operations, Cassandra gets values from the mem-table and checks the bloom filter to find the appropriate SSTable which contains the required data.
- There are three types of read request that is sent to replicas by coordinators.
  - Direct request
  - Digest request
  - Read repair request

Read Operation

- The coordinator sends direct request to one of the replicas. After that, the coordinator sends the digest request to the number of replicas specified by the consistency level and checks if the returned data is an updated data.
- After that, the coordinator sends digest request to all the remaining replicas.
- If any node gives out of date value, a background read repair request will update that data. This process is called read repair mechanism.

Read Operation



Data Model

- The data model of Cassandra is significantly different from what we normally see in an RDBMS.
- Cassandra database is distributed over several machines that operate together.
- The outermost container is known as the Cluster. For failure handling, every node contains a replica, and in case of a failure, the replica takes charge.
- Cassandra arranges the nodes in a cluster, in a ring format, and assigns data to them.

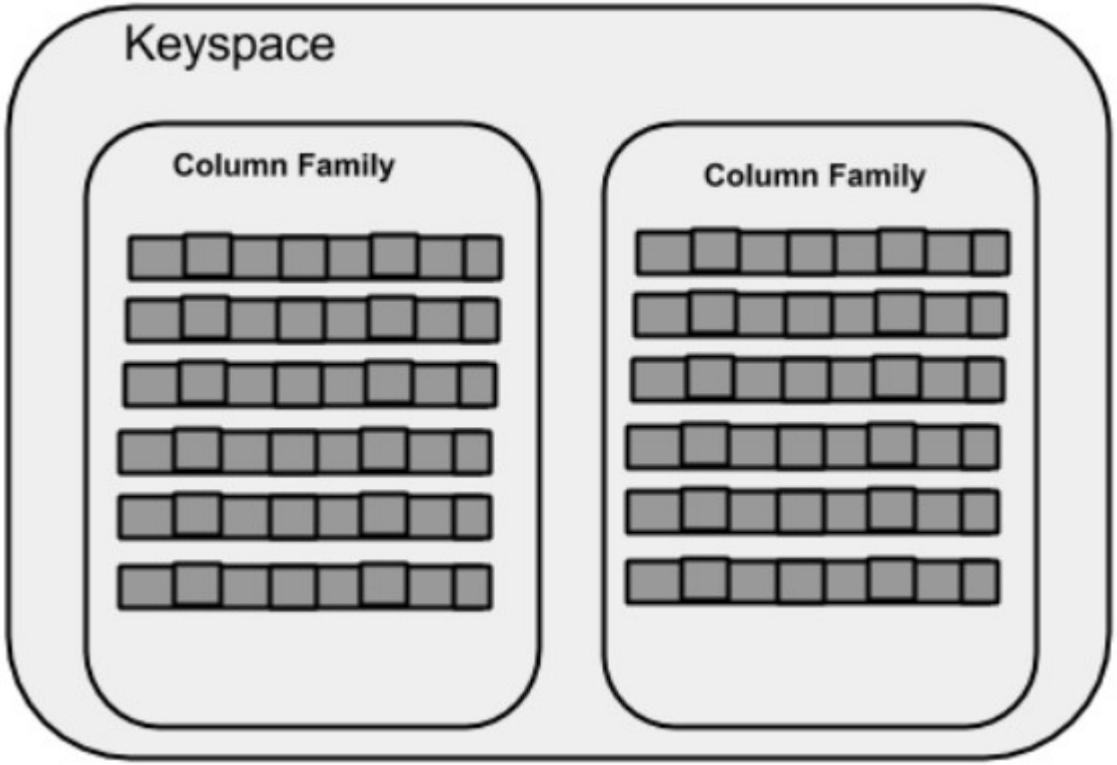
Data Model

- Keyspace is the outermost container for data in Cassandra. The basic attributes of a Keyspace in Cassandra are:
- Replication factor:
  - It is the number of machines in the cluster that will receive copies of the same data.
- Replica placement strategy:
  - It is nothing but the strategy to place replicas in the ring. We have strategies such as simple strategy (rack-aware strategy), old network topology strategy (rack-aware strategy), and network topology strategy (datacenter-shared strategy).
- Column families:
  - Keyspace is a container for a list of one or more column families. A column family, in turn, is a container of a collection of rows. Each row contains ordered columns. Column families represent the structure of your data. Each keyspace has at least one and often many column families.

Creating keyspace

- ```
  CREATE KEYSPACE Keyspace name WITH
  replication = {'class':
  'SimpleStrategy',
  'replication_factor' : 3};
  ```
  ```

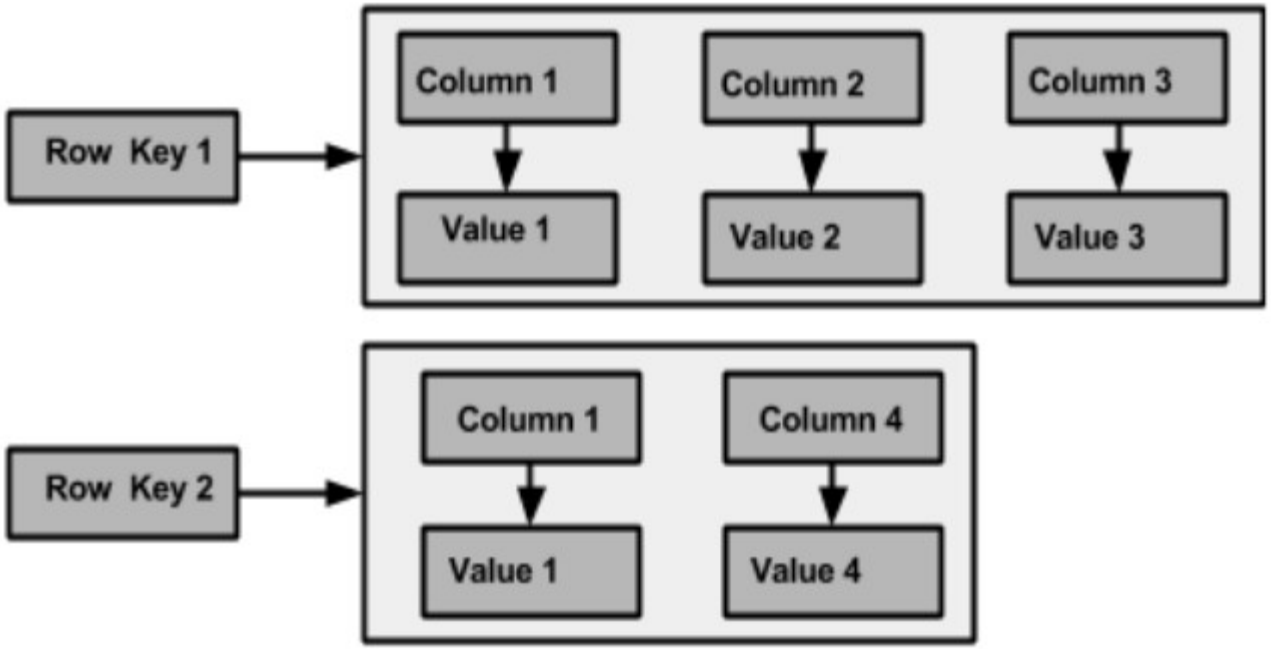
Keyspace



Column family

- A column family is a container for an ordered collection of rows. Each row, in turn, is an ordered collection of columns.
- A Cassandra column family has the following attributes:
  - keys_cached It represents the number of locations to keep cached per SSTable.
  - rows_cached It represents the number of rows whose entire contents will be cached in memory.
  - preload_row_cache: It specifies whether you want to pre-populate the row cache.

Column family



Cassandra: Use Cases

- Messaging
  - Cassandra is a great database which can handle a big amount of data. So it is preferred for the companies that provide Mobile phones and messaging services. These companies have a huge amount of data, so Cassandra is best for them.
- Handle high speed Applications
  - Cassandra can handle the high speed data so it is a great database for the applications where data is coming at very high speed from different devices or sensors.

Cassandra: Use Cases

- Product Catalogs and retail apps
  - Cassandra is used by many retailers for durable shopping cart protection and fast product catalog input and output.
- Social Media Analytics and recommendation engine
  - Cassandra is a great database for many online companies and social media providers for analysis and recommendation to their customers.

Cassandra vs MongoDB: Data Model

- MongoDB
  - MongoDB has a rich and expressive data model. This data model is ‘object-oriented’ or ‘data-oriented’. This data model can easily represent any data structure in the domain of the user. In this, data can have properties and can be nested in each other for multiple levels.
- Cassandra
  - It is more of a traditional model with table structure including rows and columns. In this data model, data is more structured and each column has a specific type.
  - This type is entered during the creation of the table. When we compare both the models, MongoDB tends to provide a rich data model.

Cassandra vs MongoDB: Master Node

- MongoDB
  - In MongoDB, there is only one master node in a cluster. This master node control a number of slave nodes. During a failure, when the master node goes down, one of the slave nodes is elected as master.
  - This process takes about 10 to 30 seconds. During this delay time, the cluster is down and cannot take any input.
- Cassandra
  - It has multiple master nodes in a cluster. Because of these multiple nodes, if one goes down, another takes its place.
  - Therefore, there is no effect on the cluster. Hence, the cluster is always available. Now, after comparing them both, Cassandra has higher availability than MongoDB.

Cassandra vs MongoDB: Seconday Index

- MongoDB
  - It has secondary indexes are a first-class construct. Because of this, it is very easy to index any property of the data stored in the database. This property makes it easy to query.
- Cassandra
  - It has cursory support for the secondary index. These indexes are limited to single columns and equality comparisons. If the application needs secondary indexes and needs flexibility in the query model, then MongoDB is better than Cassandra.

Cassandra vs MongoDB: Master Node

- MongoDB
  - In MongoDB, there is only one master node. This master node only accepts the input. Apart from this, all the nodes are used as an output. Therefore, if the data has to be written in the slave nodes, it has to pass through the master node.
- Cassandra
  - Cassandra has multiple master nodes. To input data in the other nodes, these master nodes are used. Therefore, the more master nodes a cluster has, the better it will scale. After comparing the both, Cassandra has higher scalability than MongoDB.

Cassandra vs MongoDB: QL

- MongoDB
  - MongoDB, as of now, has no support for a query language. In MongoDB, the queries are structured as JSON fragments.
- Cassandra
  - Cassandra has its own query language. This query language is CQL. This query language or CQL is very similar to SQL.
  - Hence, if we compare both on the basis of a query language, Cassandra is better than MongoDB.

Cassandra vs MongoDB: Aggregation

- MongoDB
  - MongoDB has a built-in Aggregation framework. This framework is used to run an ETL pipeline to transform the data stored in the database.
  - This framework supports small and medium data traffic. Also, with increase in the complication, the framework start facing difficulties to debug.
- Cassandra
  - Cassandra does not have a built-in aggregation framework. Cassandra uses external tools like Apache Spark, Hadoop etc. After comparing the two, MongoDB is better than Cassandra when it comes to built-in aggregation framework.

Cassandra vs MongoDB: Schema

- MongoDB
  - In MongoDB, a user has the ability to alter the enforcement of any schema on the database. Each database can be a different structure. It depends on the program or the application to interpret the data.
- Cassandra
  - Cassandra provides static typing. The user needs to define the type of the column in the beginning.

Cassandra vs MongoDB

- Database Model
  - The performance according to database depends on the schemas. Some of the schemas work best in MongoDB and some in Cassandra.
- Load Characteristics
  - When it comes to heavy data load input, Cassandra is better than MongoDB. On the other hand, when it comes to heavy data load output, both are similar.

Installation

- Operating System: Ubuntu 20.04 LTS +

CQLSH

- Cassandra CQLsh stands for Cassandra CQL shell. CQLsh specifies how to use Cassandra commands.
- After installation, Cassandra provides a prompt Cassandra query language shell (cqlsh).
- It facilitates users to communicate with it.

CQLSH: Commands

- help
  - This command is used to show help topics about the options of CQLsh commands.
- version
  - it is used to see the version of the CQLsh you are using.
- color
  - it is used for colored output.
- debug
  - It shows additional debugging information.
- execute
  - It is used to direct the shell to accept and execute a CQL command.

CQLSH: Commands

- file= "file name"
  - By using this option, cassandra executes the command in the given file and exits.
- no-color
  - It directs cassandra not to use colored output.
- u "username"
  - Using this option, you can authenticate a user. The default user name is: cassandra.
- p "password"
  - Using this option, you can authenticate a user with a password. The default password is: cassandra.

Thank you

@mitu_skillologies

@mITuSkillologies

@mitu_group

@mitu-skillologies

@MITUSkillologies

kaggle

@mituskillologies

Web Resources
https://mitu.co.in
http://tusharkute.com

@mituskillologies

contact@mitu.co.in

tushar@tusharkute.com