Apache Cassandra

- **Introduction**

  Apache Cassandra is a free and open source distributed NoSQL database management system designed to handle large amounts of structured data spread out across many commodity servers. Cassandra offers robust support for clusters spanning multiple datacenters[1] with asynchronous master less replication allowing low latency operations for all clients.

  It is a solution developed by Facebook when they run the largest social networking platform that serves hundreds of millions users at peak times using tens of thousands of servers located in many data centers around the world. [2] Later, it becomes a open source and used by many other companies.

  The motivation behind Cassandra based on the understanding that system and hardware failures can and do occur. Cassandra addresses the problem of failures by employing a peer-to-peer distributed system across homogeneous nodes where data is distributed among all nodes in the cluster. Each node frequently exchanges state information about itself and other nodes across the cluster using peer-to-peer gossip communication protocol. A sequentially written commit log on each node captures write activity to ensure data durability. [2]

- **Data model**

  A table in Casandra is a distributed multi-dimensional map indexed by a key. The value is an object which is highly structured. Columns are grouped into sets called column families.

  To explain it in more specific way. A column family (called "table" since CQL 3) resembles a table in an RDBMS. Column families contain rows and columns. Each row is uniquely identified by a row key. Each row has multiple columns, each of which has a name, value, and a timestamp. Unlike a table in an RDBMS, different rows in the same column family do not have to share the same set of columns, and a column may be added to one or multiple rows at any time.

  As we have mentioned, Cassandra is a NoSQL system, as alternative, we use CQL (Cassandra Query Language). CQL is a simple interface for accessing Cassandra. CQL adds an abstraction layer that hides implementation details of this structure and provides native syntaxes for collections and other common encodings. Language drivers are available for Java (JDBC), Python (DBAPI2), Node.JS (Helenus), Go (gocql) and C++. [3]

  The Cassandra API consists of the following three simple methods:

insert (table, key, rowMutation)

get(table,key,columnName)

delete(table,key,columnName) [2]

Tables may be created, dropped, and altered at run-time without blocking updates and queries. In addition, Cassandra's architecture allows any authorized user to connect to any node in any datacenter and access data.

- **Writing and reading data**

  Client read or write requests can be sent to any node in the cluster. When a client connects to a node with a request, that node serves as the coordinator for that particular client operation. The coordinator acts as a proxy between the client application and the nodes that own the data being requested. The coordinator determines which nodes in the ring should get the request based on how the cluster is configured.

  - Writing data
    Cassandra processes data at several stages on the write path, starting with the immediate logging of a write and ending in with a write of data to disk:
    - Logging data in the commit log
    - Writing data to the memtable
    - Flushing data from the memtable
    - Storing data on disk in SSTables

    Here The memtable is a write-back cache of data partitions that Cassandra looks up by key. The memtable stores writes in sorted order until reaching a configurable limit, and then is flushed. To flush the data, Cassandra writes the data to disk, in the memtable-sorted order. A partition index is also created on the disk that maps the tokens to a location on disk. [4]

  - Reading data
    To satisfy a read, Cassandra must combine results from the active memtable and potentially multiple SSTables. Cassandra processes data at several stages on the read path to discover where the data is stored, starting with the data in the memtable and finishing with SSTables:
    - Check the memtable
    - Check row cache, if enabled
    - Checks Bloom filter

- Checks partition key cache, if enabled
- Goes directly to the compression offset map if a partition key is found in the partition key cache, or checks the partition summary if not
- If the partition summary is checked, then the partition index is accessed
- Locates the data on disk using the compression offset map
- Fetches the data from the SSTable on disk. [5]

- **Main Features**

  - Fault tolerant
    Data is automatically replicated to multiple nodes for fault-tolerance. Replication across multiple data centers is supported. Failed nodes can be replaced with no down time.

  - Supports replication and multi data center replication
    Cassandra uses replication to achieve high availability and durability. Each data item is replicated at N hosts, where N is the replication factor configured “per-instance”. Replication strategies are configurable.[6] Cassandra is designed as a distributed system, for deployment of large numbers of nodes across multiple data centers. Key features of Cassandra’s distributed architecture are specifically tailored for multiple-data center deployment, for redundancy, for failover and disaster recovery.

  - Decentralized
    There are no single points of failure. Every node in the cluster has the same role. Every node in the cluster is identical. Data is distributed across the cluster (so each node contains different data), but there is no master as every node can service any request.

  - Scalability
    Read and write throughput both increase linearly as new machines are added, with no downtime or interruption to applications.
    Some of the largest production deployments include Apple's, with over 75,000 nodes storing over 10 PB of data, Netflix (2,500 nodes, 420 TB, over 1 trillion requests per day), Chinese search engine Easou (270 nodes, 300 TB, over 800 million requests per day), and eBay (over 100 nodes, 250 TB). [7]

- **Reference**
  [1] Casares, Joaquin (2012-11-05). "Multi-datacenter Replication in Cassandra". DataStax. Retrieved 2013-07-25. Cassandra’s innate datacenter concepts are important as they allow multiple workloads to be run across multiple datacenters…
  [2] Avinash Lakshman , Prashant Malik. Cassandra - A Decentralized Structured Storage System
  [3] "DataStax C/C++ Driver for Apache Cassandra". DataStax. Retrieved 15

December 2014.
[4]http://docs.datastax.com/en/cassandra/3.0/cassandra/dml/dmlHowDataWritten.html.
[5] http://docs.datastax.com/en/cassandra/3.0/cassandra/dml/dmlAboutReads.html.
[6] "Deploying Cassandra across Multiple Data Centers". DataStax. Retrieved 11 December 2014.
[7] http://cassandra.apache.org/