Cassandra - Cluster, Datacenters, Racks and Nodes

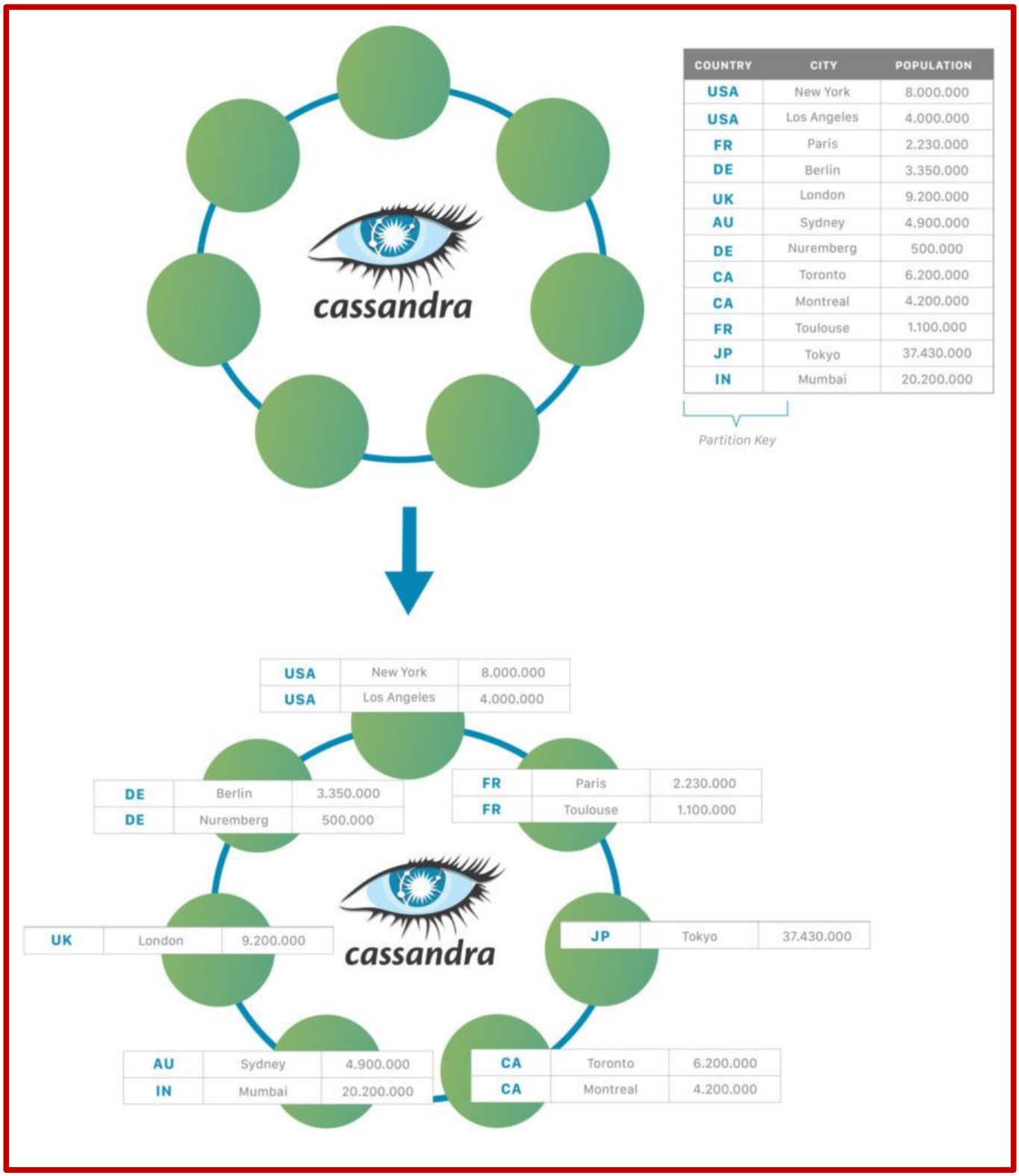
1. Introduction

In this article, we'll have a close look at Cassandra's architecture. We'll find out about data storing in a distributed architecture, and we'll discuss basic architecture components.

2. Cassandra Overview

Apache Cassandra is a NoSQL, distributed database management system. **The main advantage of Cassandra is that it can handle a high volume of structured data across commodity servers. Moreover, it provides high availability and provides no single point of failure.** Cassandra achieves this by using a ring-type architecture, where the smallest logical unit is a node. It uses the partitioning of data for optimizing queries.

Every piece of data has a partition key. The partition key for every row is hashed. As a result, we'll get a unique token for every piece of data. For each node, there is an assigned range of tokens. Consequently, data with the same token is stored on the same node. The ring architecture of nodes is shown below:



3. Cassandra Components

3.1. Node

A Node is the basic infrastructure component of Cassandra. **It's a fully functional machine that connects with other nodes in the cluster through the high internal network.**

The name of this network is *Gossip Protocol*. To clarify, the machine can be a physical server or an EC2 instance, or a virtual machine. All nodes are organized with ring network topology. Importantly, every node is independent and has the same role in the ring. Cassandra arranges nodes in a *peer-to-peer structure.* The node contains the actual data.

Each node in a cluster can accept read and write requests. Therefore, it doesn't matter where the data is actually located in the cluster. We'll always get the newest version of data.

3.2. Virtual Node

Newer versions of Cassandra use virtual nodes or *vnodes* for short. **A virtual node is the data storage layer within a server.**

There are 256 virtual nodes per server by default. As we discussed in the previous paragraph, each node has a range of tokens assigned. Every virtual node uses a sub-range of tokens from the node they belong to.

These virtual nodes provide greater flexibility in the system. Consequently, it's easier for Cassandra to add new nodes to the cluster when we need them. When our data has unequally distributed tokens between nodes, we can easily extend the storage capacity by extending virtual nodes to the more loaded node.

3.4. Server

When we use the term *server*, we'll mean a machine with the Cassandra software installed. Every node has a single instance of Cassandra, which is technically a server. As we said earlier, each instance of Cassandra has evolved to contain 256 virtual nodes. The Cassandra server runs core processes. For example, processes like spreading replicas around nodes or routing requests.

3.5. Rack

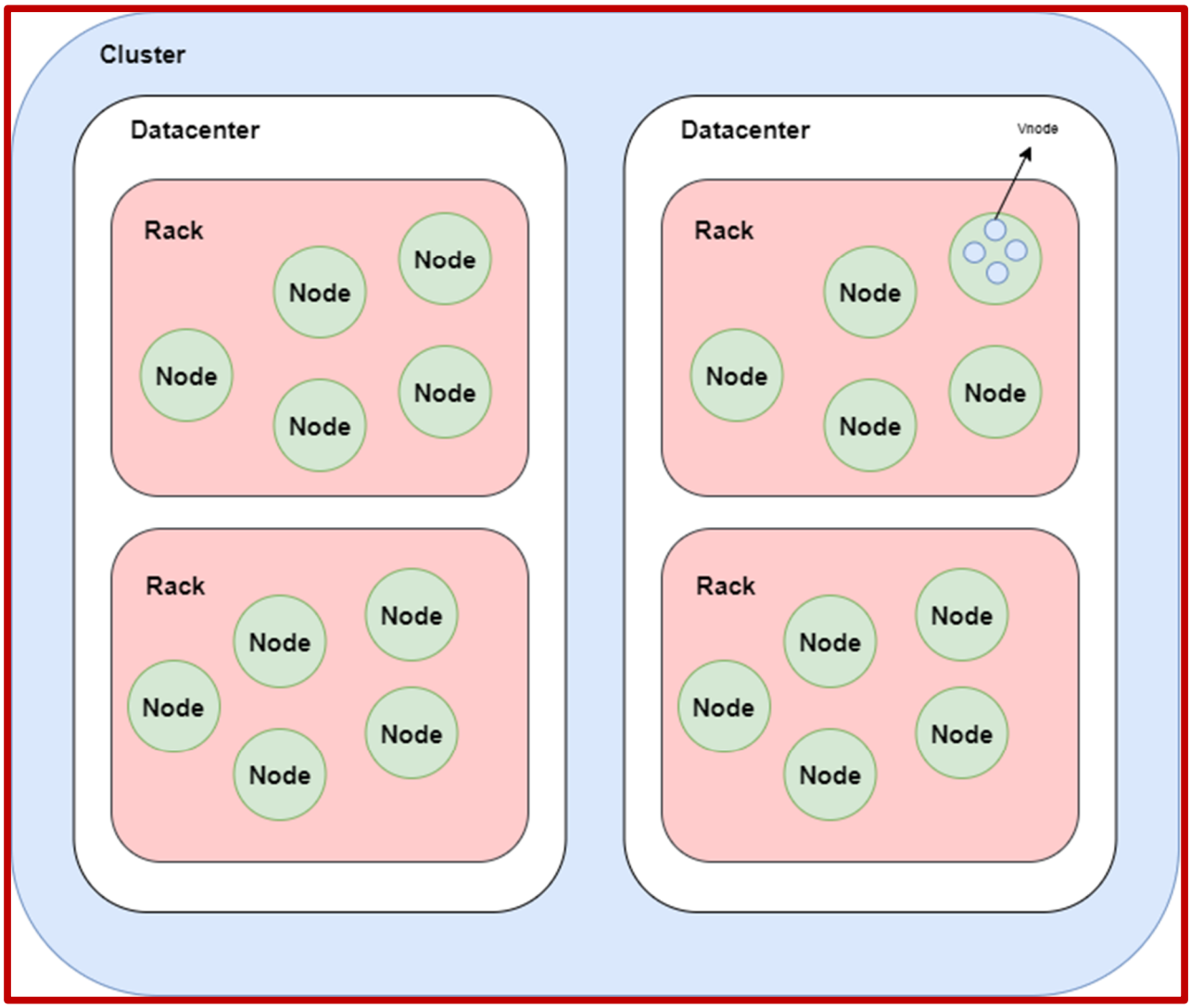
A Cassandra rack is a logical grouping of nodes within the ring. In other words, a rack is a collection of servers. The database uses racks so that it can ensure replicas are distributed among different logical groupings. As a result, it can send operations not only to just one node. Multiple nodes, each on a separate rack, can provide greater fault tolerance and availability.

3.6. Datacenters

A datacenter is a logical set of racks. The datacenter should contain at least one rack. We can say that the Cassandra Datacenter is a group of nodes related and configured within a cluster for replication purposes. So, it helps to reduce latency, prevent transactions from impact by other workloads and related effects. What's more, the replication factor can also be set up to write to multiple datacenters. As a result, Cassandra can provide additional flexibility in architectural design and organization.

3.7. Cluster

A cluster is a component that contains one or more datacenters. It's the most outer storage container in the database. One database contains one or more clusters. The hierarchy of elements in the Cassandra cluster is:



First, we have clusters that consist of datacenters. Inside of datacenters, we have nodes that contain by default 256 virtual nodes.

4. Data Replication

Now, when we know the basic components of Cassandra. Let's talk about how Cassandra manages data around its structure. Some systems can not allow for data loss or interruption in data delivery. The solution is to provide a backup when the problem has occurred. For example, it can be hardware problems, or links can be down at any time during the data process. Cassandra stores data replicas on multiple nodes to ensure reliability and fault tolerance.

5.1. Replication Factor

We can determine the number of replicas and their location by the replication factor and replication strategy. **The replication factor is the total number of replicas across the cluster**. When we set this factor to one, it means that only one copy of each row exists in a cluster and so on**.** We can set this factor on the datacenter level and on rack level.

5.1. Replication Strategy

The replication strategy controls how the replicas are chosen. The importance of replicas is the same. Cassandra has two strategies for determining which nodes contain replicated data. The first one is called the *SimpleStrategy,* and it is unaware of the logical division of nodes for datacenters and racks. The second one is *NetworkTopologyStrategy* is more complicated and is both racks aware and datacenter aware. We can define how many replicas would be placed in different datacenters by using The *NetworkTopologyStrategy*. Additionally, it tries to avoid situations when two replicas are placed on the same rack.

5. Conclusion

This article introduces the basic components of Cassandra's architecture. We covered the key concepts of this database that ensure its high availability and partitioning tolerance. We also talked about data partitioning and data replication.