Cassandra Essentials Tutorial Series

Understanding Data Partitioning and Replication in Apache Cassandra

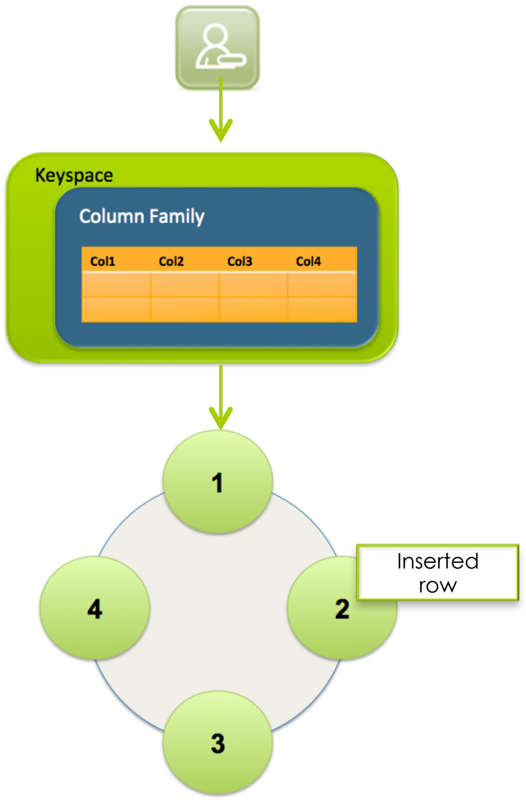
DataStax

Agenda

- Overview of partitioning
- Setting up data partitioning
- Overview of replication
- Replication strategies (e.g. single, multi-data center)
- Replication mechanics
- Where to get Cassandra

Overview of Data Partitioning in Cassandra

Cassandra is a distributed database management system that easily and transparently partitions your data across all participating nodes in a database cluster. Each node is responsible for part of the overall database.



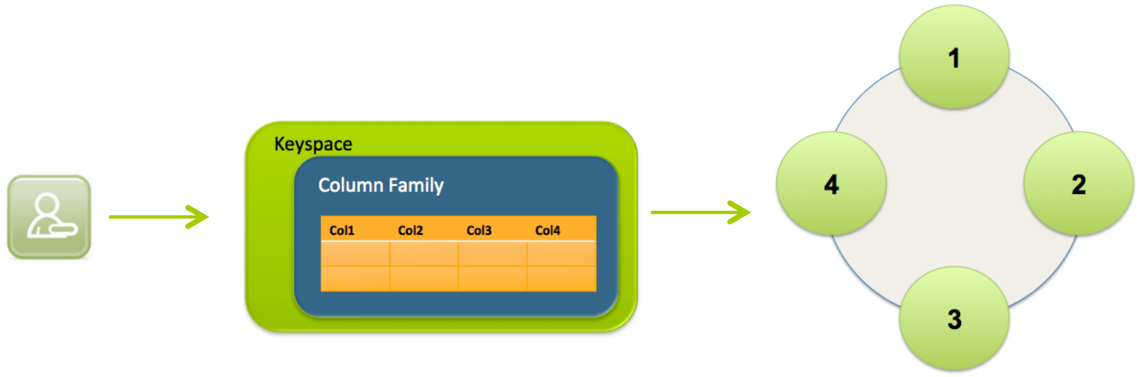
Data is inserted and assigned a row key in a column family

Data placed on node based on its column family row key

Overview of Data Partitioning in Cassandra

There are two basic data partitioning strategies:

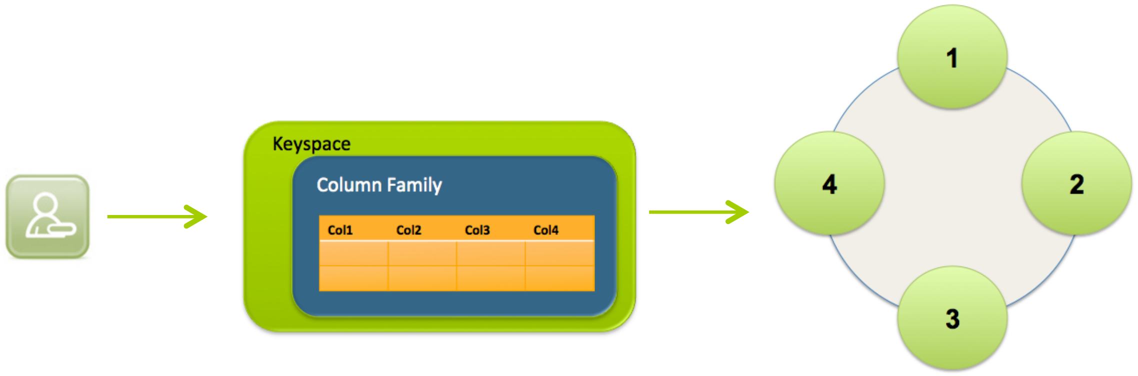
1. **Random partitioning** – this is the default and recommended strategy. Partitions data as evenly as possible across all nodes using an MD5 hash of every column family row key
2. **Ordered partitioning** – stores column family row keys in sorted order across the nodes in a database cluster



Setting up Data Partitioning in Cassandra

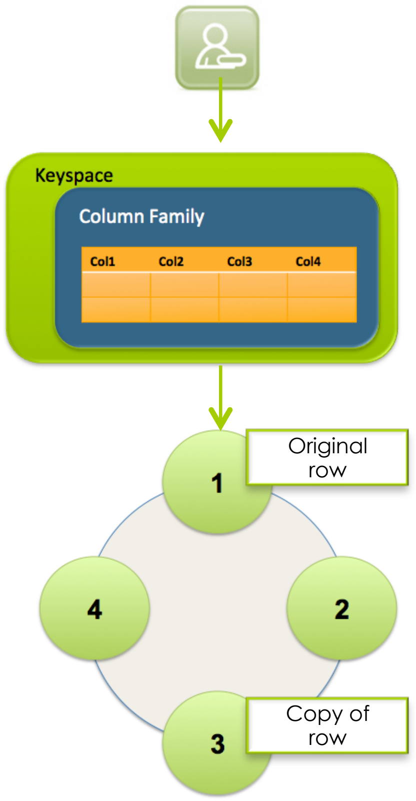
The data partitioning strategy is controlled via the Cassandra configuration file (`cassandra.yaml`) `partitioner` option. There are no other mechanics, work, sharding, etc., to partition data in Cassandra.

Note that once a cluster is initialized with a partitioner option, it cannot be changed without reloading all of the data in the cluster.



Overview of Replication in Cassandra

To ensure fault tolerance and no single point of failure, you can replicate one or more copies of every row in a column family across participating nodes in a database cluster.



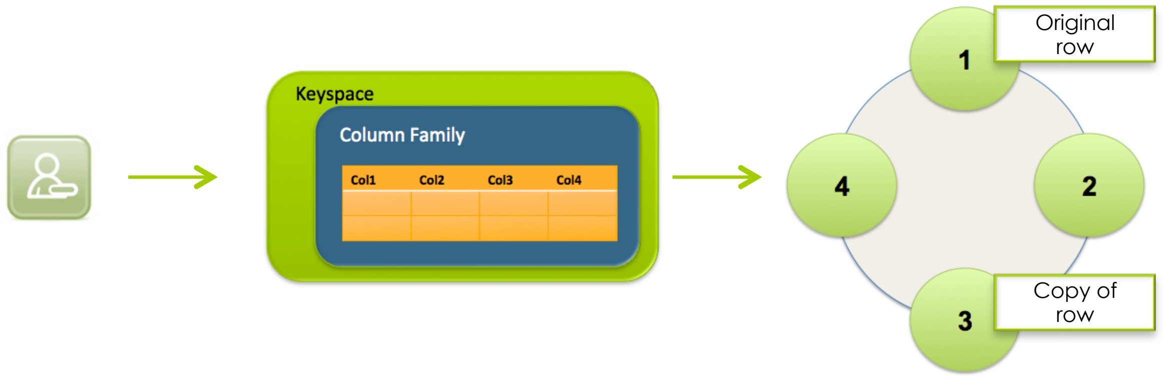
Data is inserted and assigned a row key in a column family

Copy of row is replicated across various nodes in the cluster based on the assigned replication factor

Overview of Replication in Cassandra

Replication is controlled by what is called the ***replication factor***. A replication factor of 1 means there is only one copy of a row in a cluster. A replication factor of 2 means there are two copies of a row stored in a cluster.

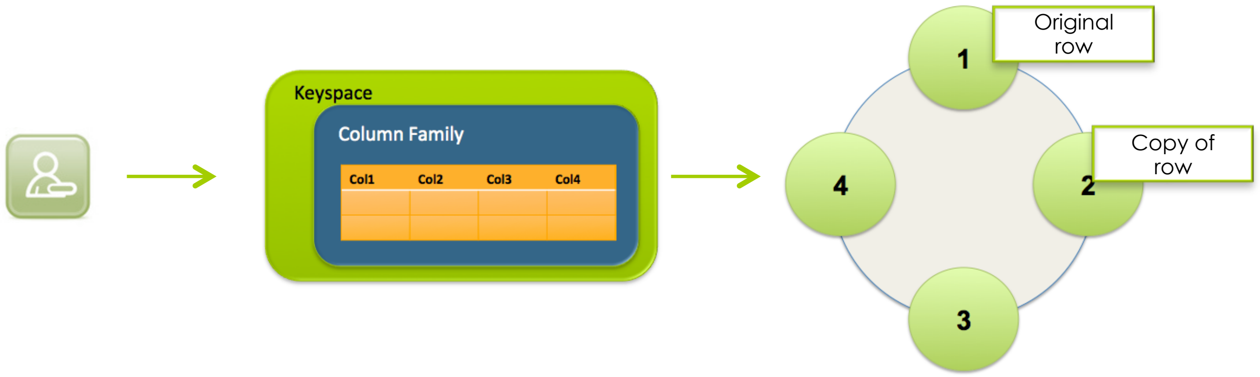
Replication is controlled at the keyspace level in Cassandra.



Replication Strategies

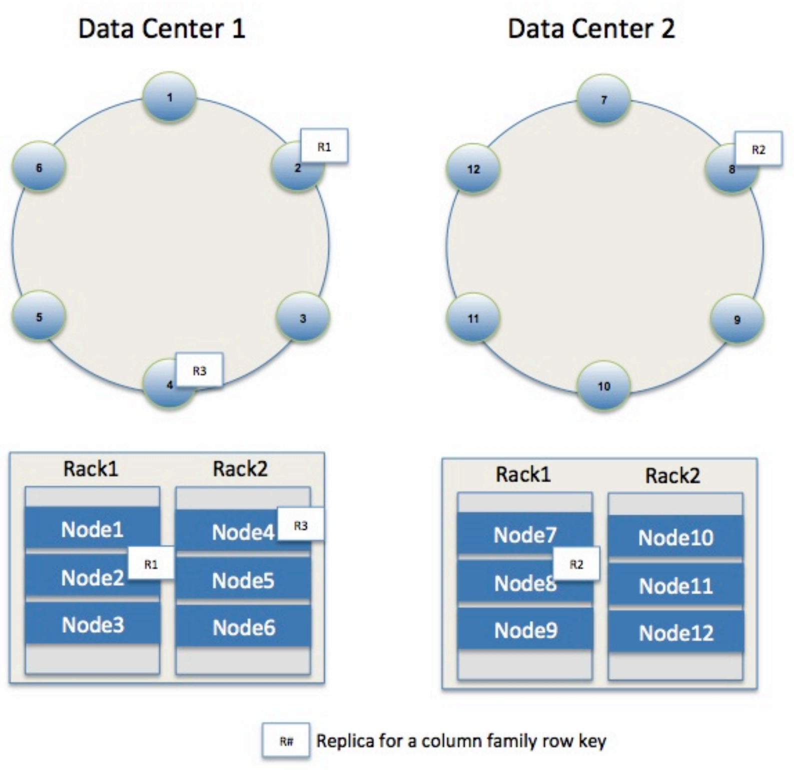
There are different replication strategies:

Simple Strategy: places the original row on a node determined by the partitioner. Additional replica rows are placed on the next nodes clockwise in the ring without considering rack or data center location.



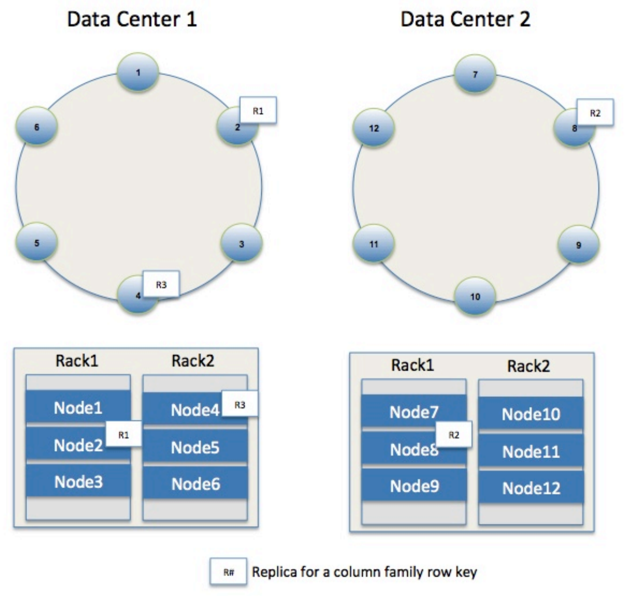
Replication Strategies

Network Topology Strategy: allows for replication between different racks in a data center and/or between multiple data centers. This strategy provides more control over where replica rows are placed.



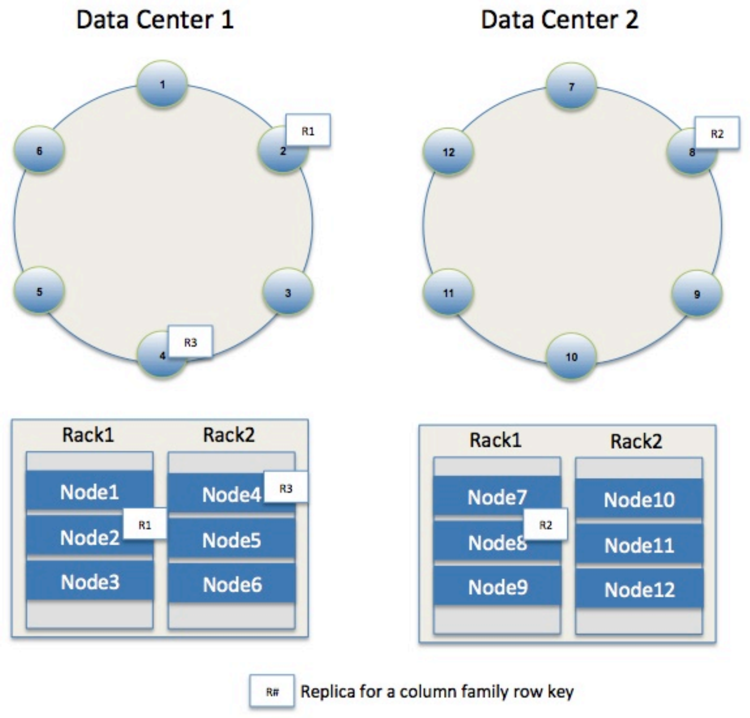
Replication Strategies

Network Topology Strategy: The original row is placed according to the partitioner. Additional replica rows in the same data center are then placed by walking the ring clockwise until a node in a different rack from the previous replica is found. If there is no such node, additional replicas will be placed in the same rack.



Replication Strategies

Network Topology Strategy: To replicate data between 1-n data centers, a **replica group** is defined and mapped to each logical or physical data center. This definition is specified when a keyspace is created in Cassandra.



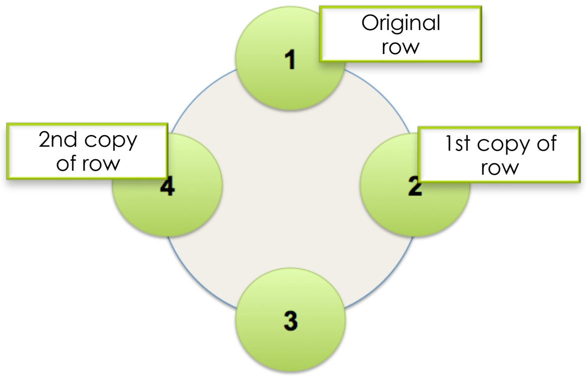
Replication Strategies

Below is a CQL example of creating a keyspace that uses the Network Topology replication strategy and has three data replicas:

```
CREATE KEYSPACE mykeyspace WITH
strategy_class = 'NetworkTopologyStrategy' AND
strategy_options:DC1 = 3;
```

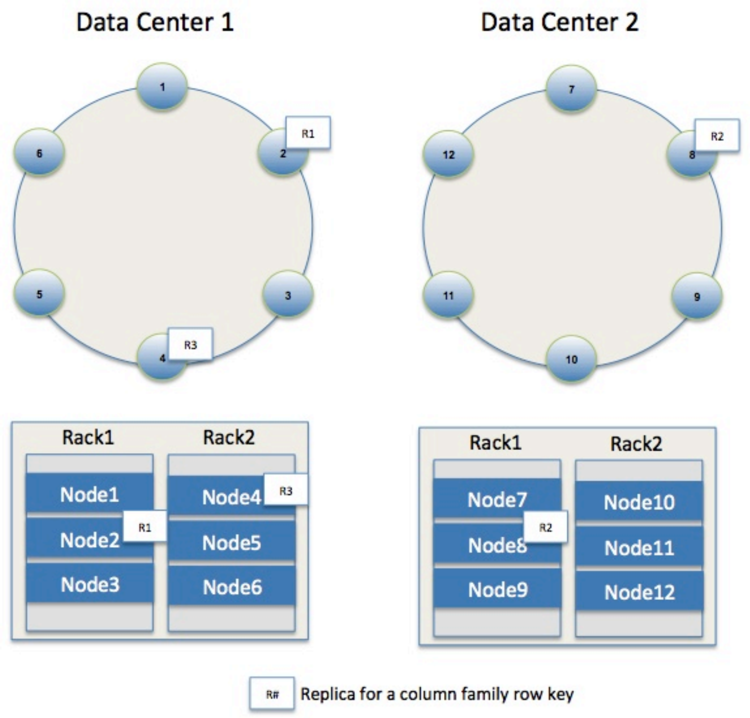
Replica group

Number of replicas



Replication Mechanics

Cassandra uses a **snitch** to define how nodes are grouped together within the overall network topology (such as rack and data center groupings). The snitch is defined in the `cassandra.yaml` file



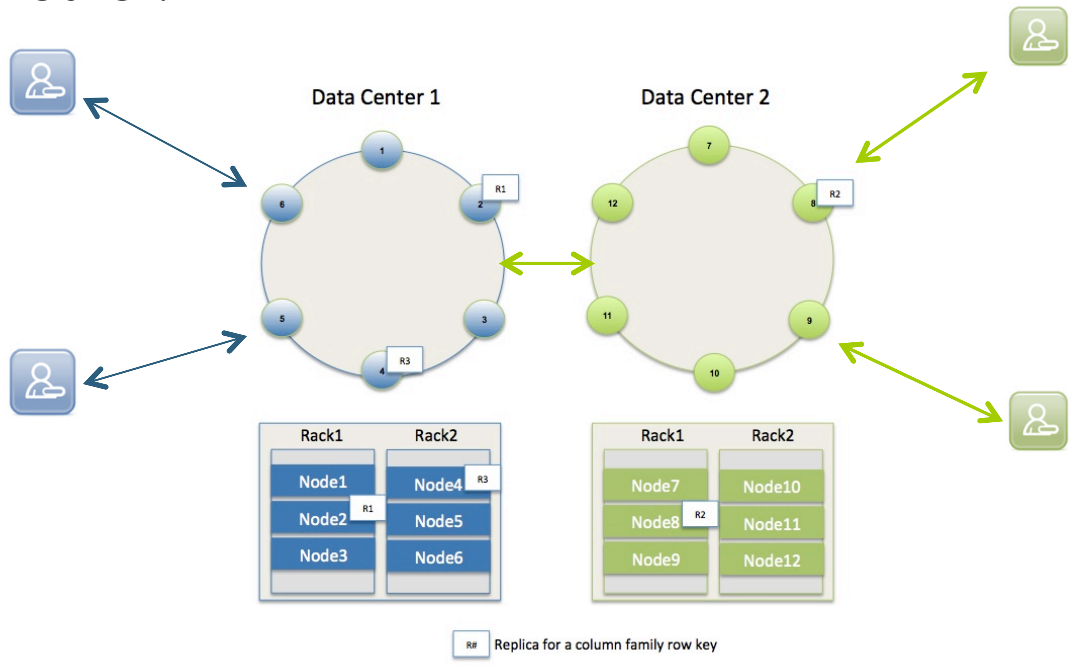
Replication Mechanics

The basic snitches include:

1. **Simple Snitch** – the default and used for the simple replication strategy
2. **Rack Inferring Snitch** - infers the topology of the network by analyzing the node IP addresses. This snitch assumes that the second octet identifies the data center where a node is located, and the third octet identifies the rack.
3. **Property File Snitch** – determines the location of nodes by referring to a user-defined description of the network details located in the property file `cassandra-topology.properties`.
4. **EC2 Snitch** - is for deployments on Amazon EC2 only. Instead of using the IP to infer node location, this snitch uses the AWS API to request region and availability zone.

Reading and Writing to Cassandra Nodes

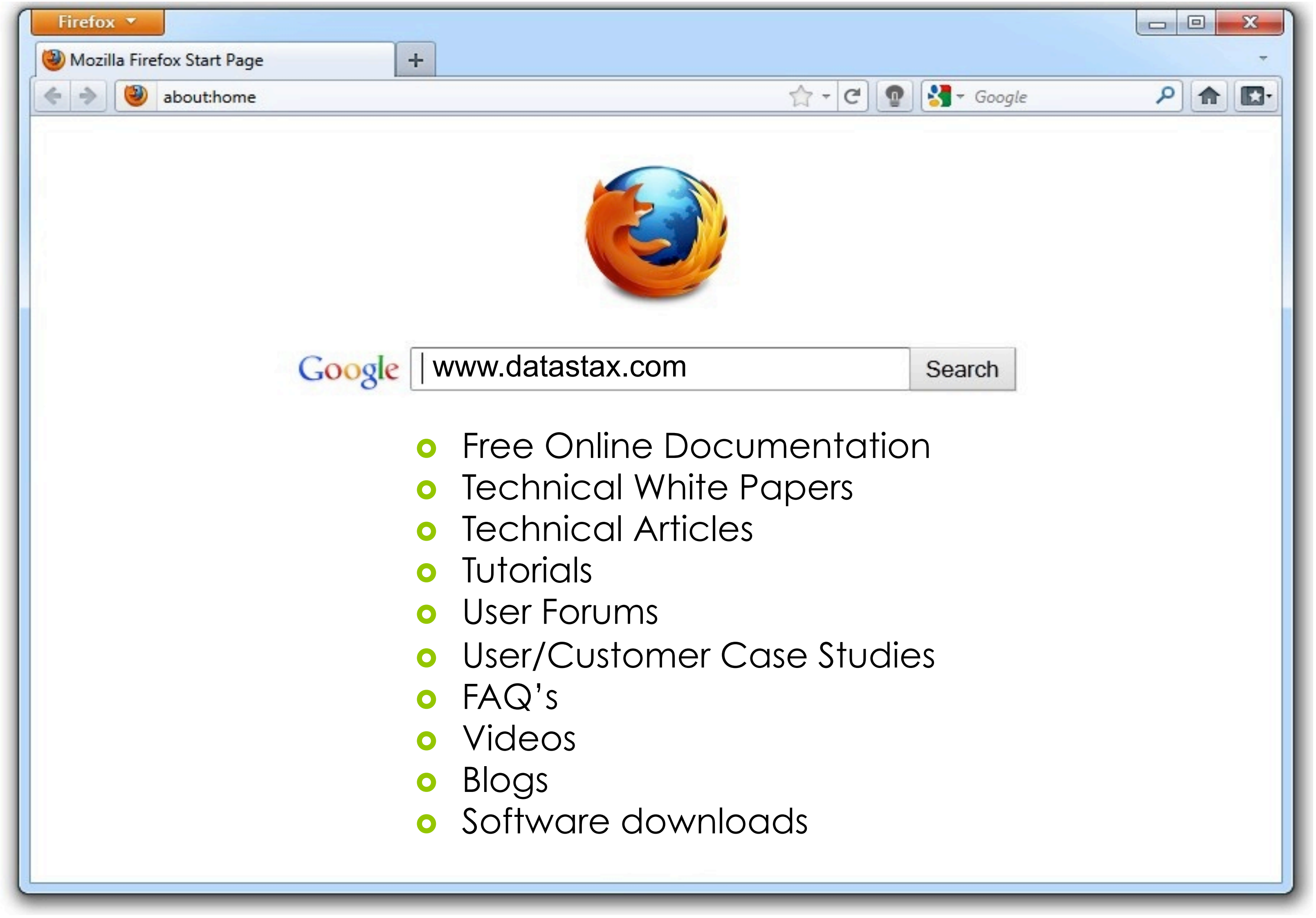
Cassandra is a read/write anywhere architecture, so any user can connect to any node in any data center and read/write the data they need, with all writes being partitioned and replicated for them automatically throughout the cluster.



Where to get Cassandra?

- Go to www.datastax.com
- DataStax makes free smart start installers available for Cassandra that include:
  - The most up-to-date Cassandra version that is production quality
  - A version of DataStax OpsCenter, which is a visual, browser-based management tool for managing and monitoring Cassandra
  - Drivers and connectors for popular development languages
  - Same database and application
  - Automatic configuration assistance for ensuring optimal performance and setup for either stand-alone or cluster implementations
  - Getting Started Guide

Where Can I Learn More?



www.datastax.com

- Free Online Documentation
- Technical White Papers
- Technical Articles
- Tutorials
- User Forums
- User/Customer Case Studies
- FAQ's
- Videos
- Blogs
- Software downloads

Cassandra Essentials Tutorial Series

Understanding Data Partitioning and Replication in Apache Cassandra

Thanks!

DataStax