

Apache Cassandra Advanced

Need : New Era of data – Big Data

Big data involves data that

- 1) is high velocity in nature;
- 2) combines structured, semi-structured, and unstructured data;
- 3) can include enormous volumes; and
- 4) typically involves complexity in data distribution and synchronization.

The massive scale, high performance, and never-go-down nature of these applications has forged a new set of technologies that have replaced the legacy RDBMS with NoSQL database like Cassandra

The Architecture of Cassandra

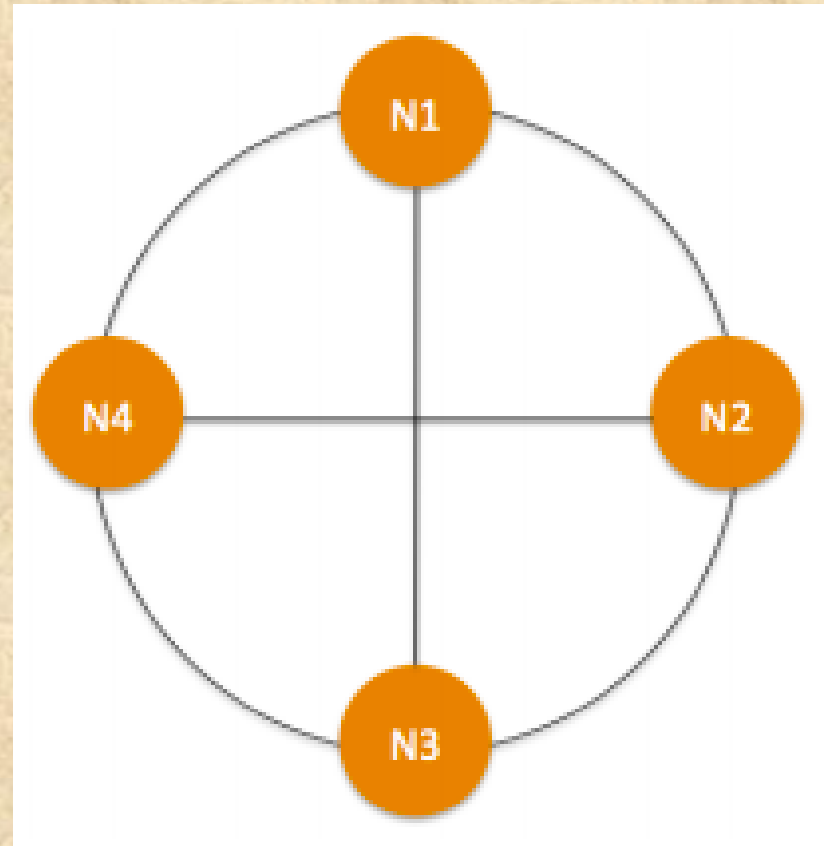
- Set of nodes
- peer-to-peer distributed architecture
- all nodes are the same; there is no concept of a master node
- with all nodes communicating with each other via a ***gossip protocol***
- capable of handling petabytes of information and thousands of concurrent users/operations per second (across multiple data centers)
- no single point of failure, true 24x7 availability

Gossip protocol

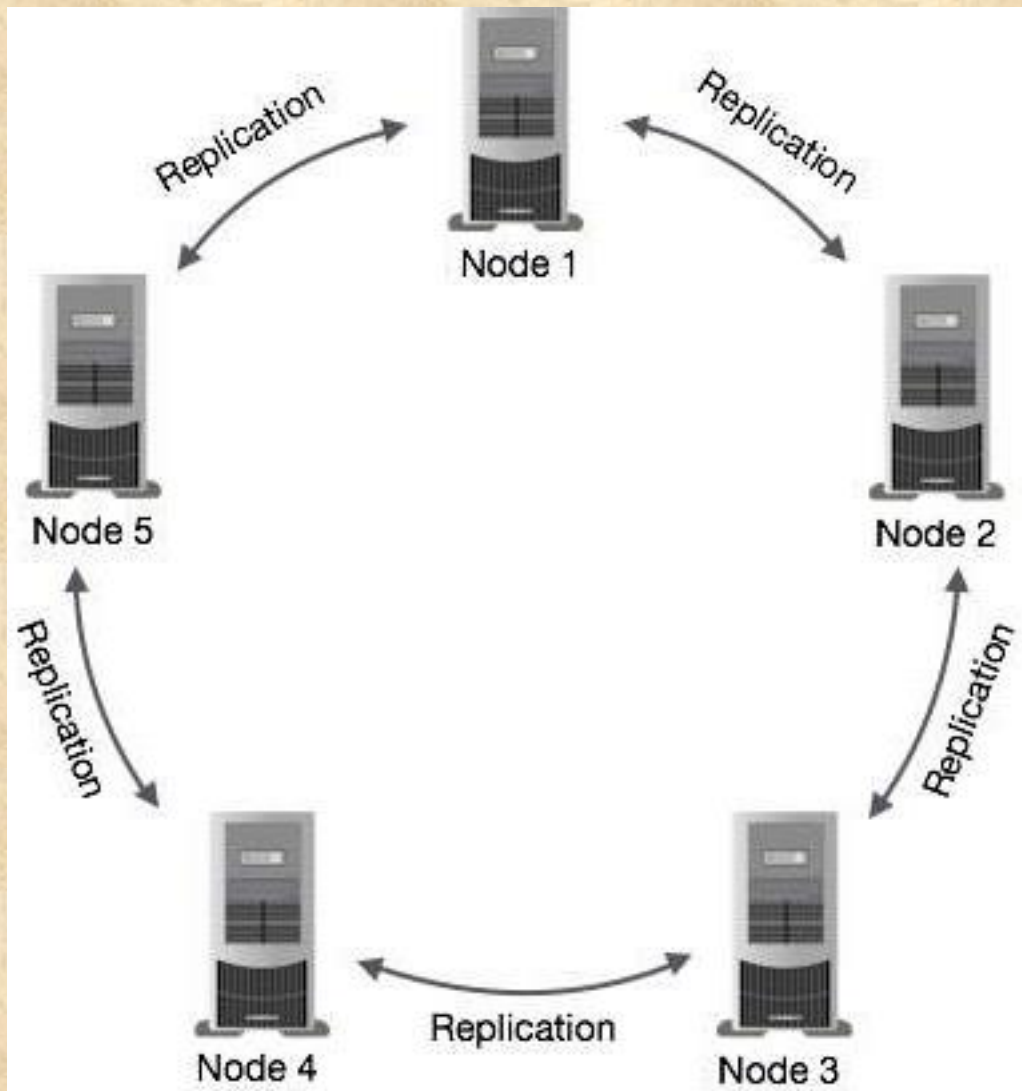
- It is similar to real-world gossip, where a node (say B) tells a few of its peers in the cluster what it knows about the state of a node (say A).
- Those nodes tell a few other nodes about A,
- and over a period of time, all the nodes know about A.

Distributing and Replicating Data

- Automatic data distribution across all nodes that participate in a “ring” or database cluster.
- Data is transparently partitioned across all nodes
 - Randomized (default)
 - ordered fashion
- Easy user-defined replication
 - “replication factor” parameter in keyspace creation

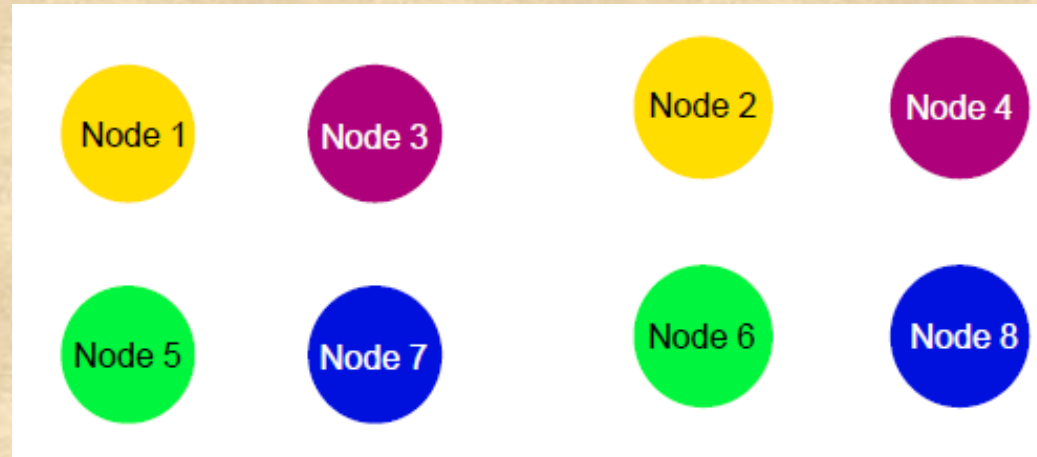


Replication

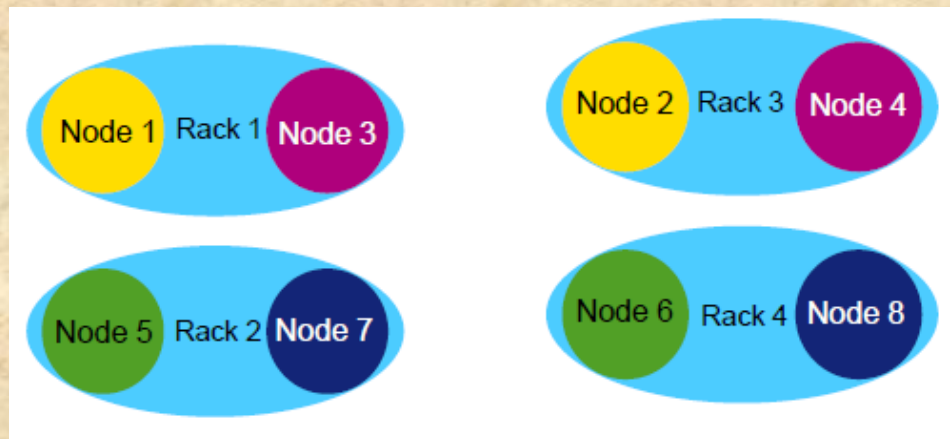


Architecture ...

- One Node : Single Cassandra instance

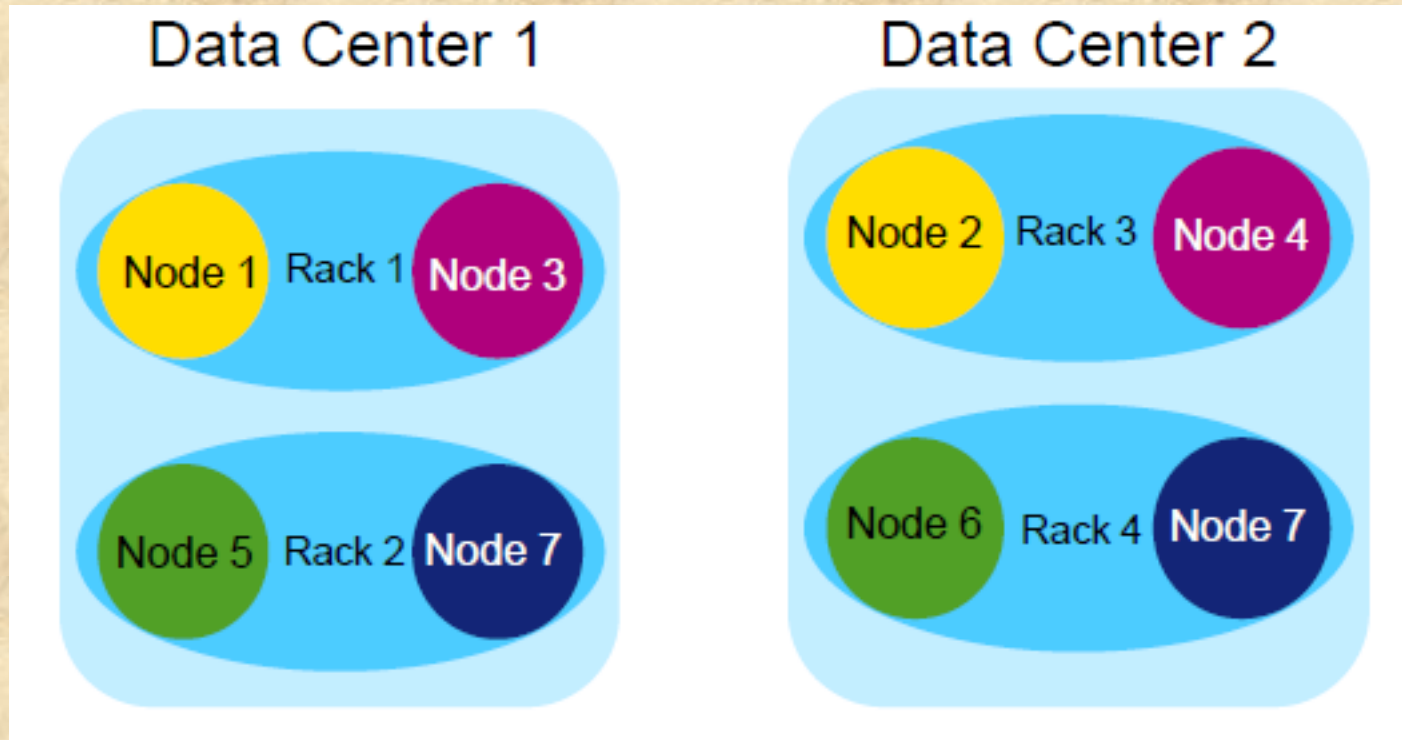


- Rack : Logical set of nodes

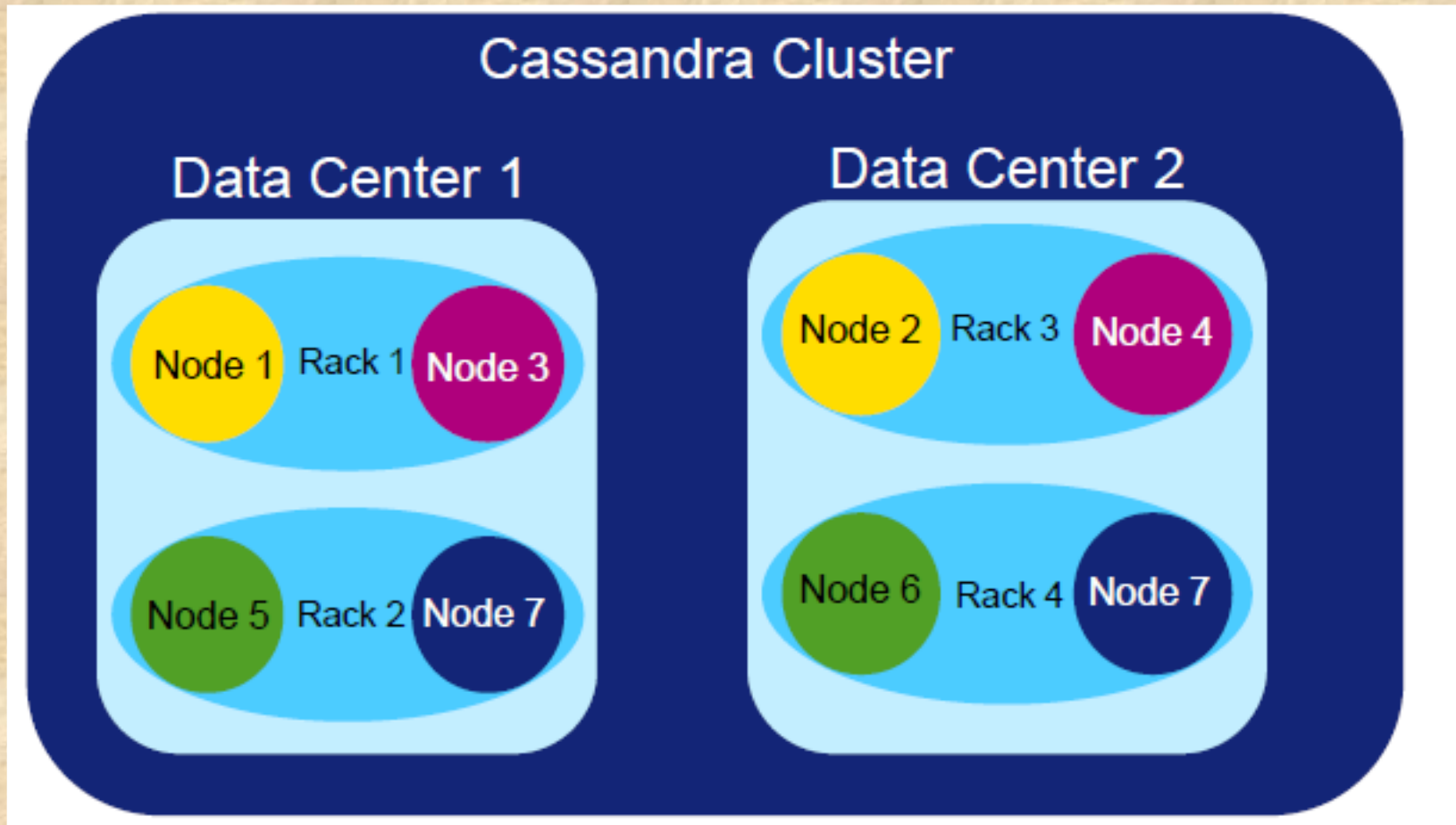


Architecture ...

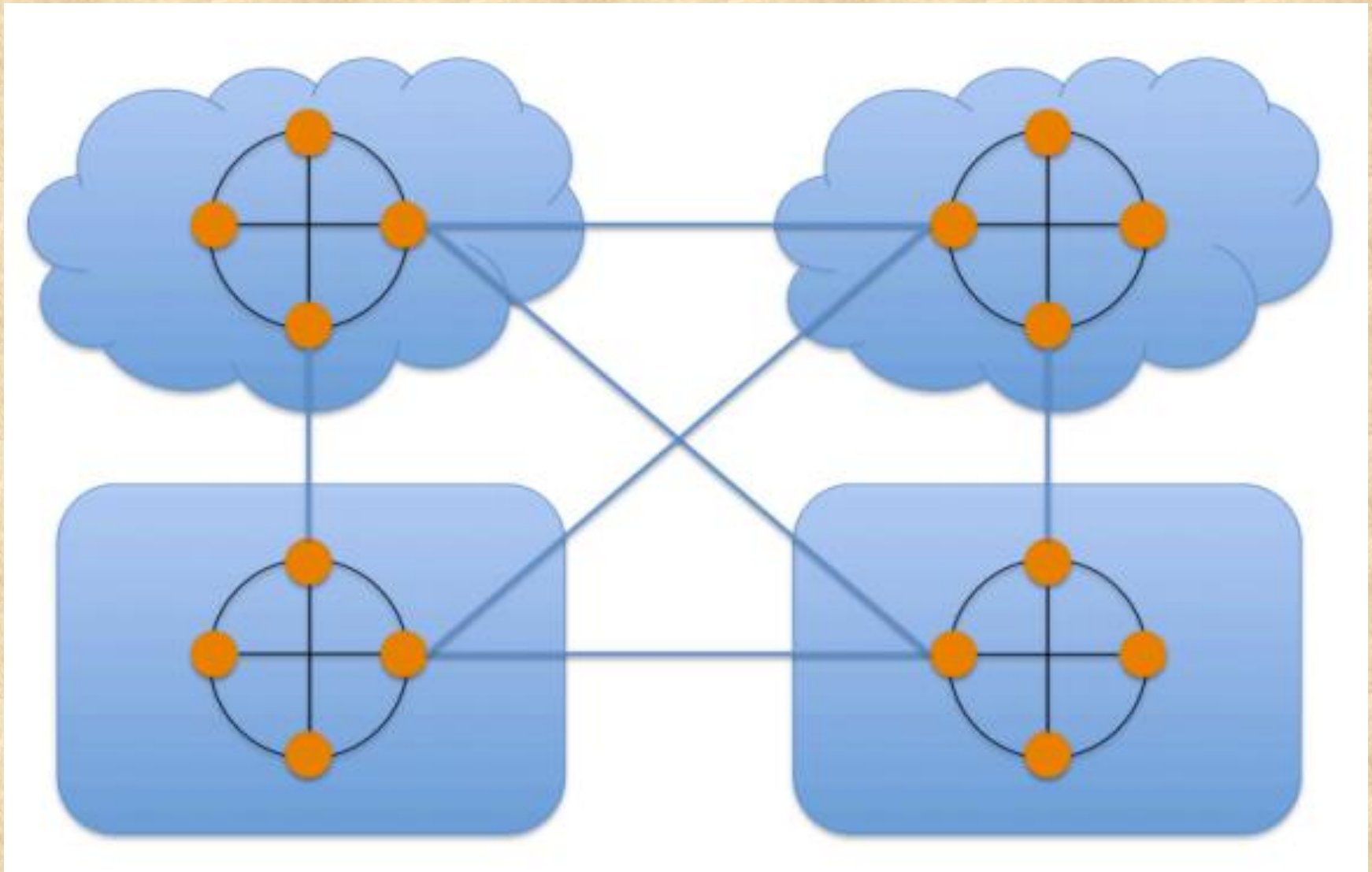
- Data Center : Logical set of Racks



Architecture ...



Multi-Data Center and Cloud Support

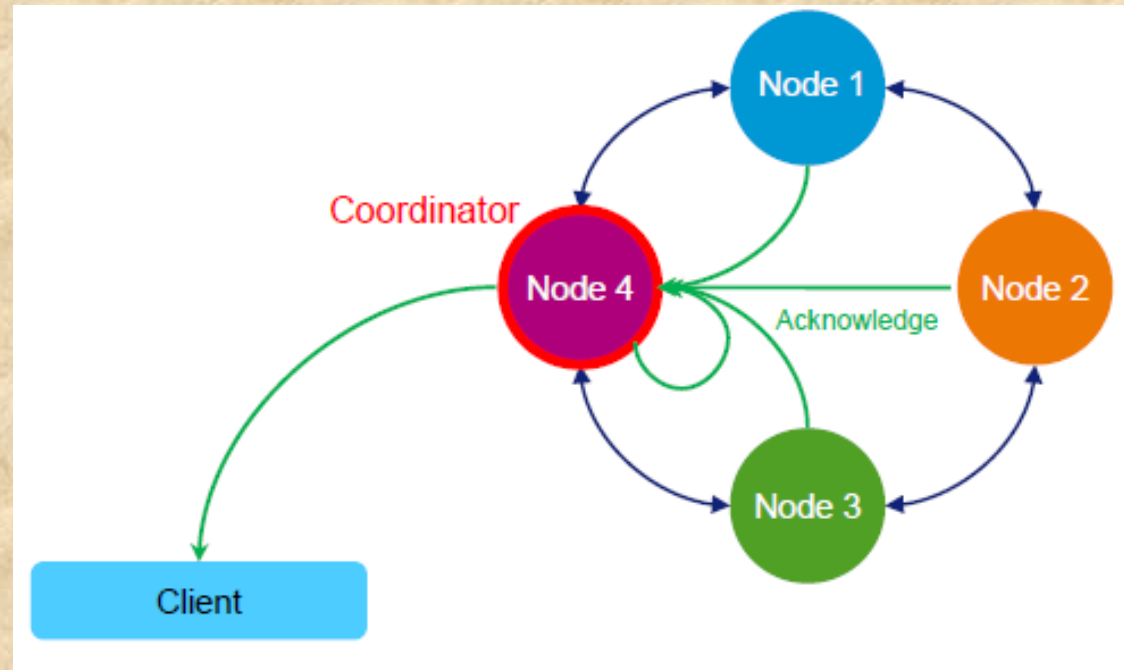


Reading and Writing Data

- true “**location independent**” architecture
- any node in a Cassandra cluster may be read or written to – ***true read/write-anywhere design.***
- it is first written to a **commit log**
- also to ***memtable***
- flushed to a disk structure called an ***sstable***
- user may requests data from any node, Cassandra engine assembled it from other nodes.

Request Coordination

- **Coordinator** : the node chosen by the client to receive a particular read or write request to its cluster.

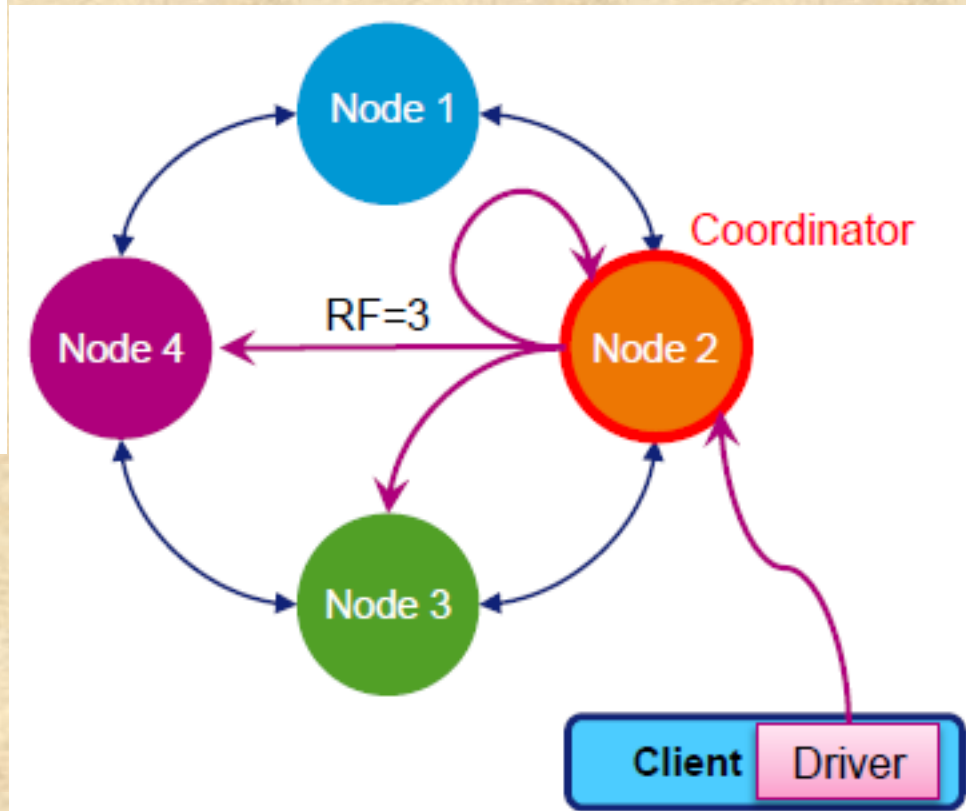
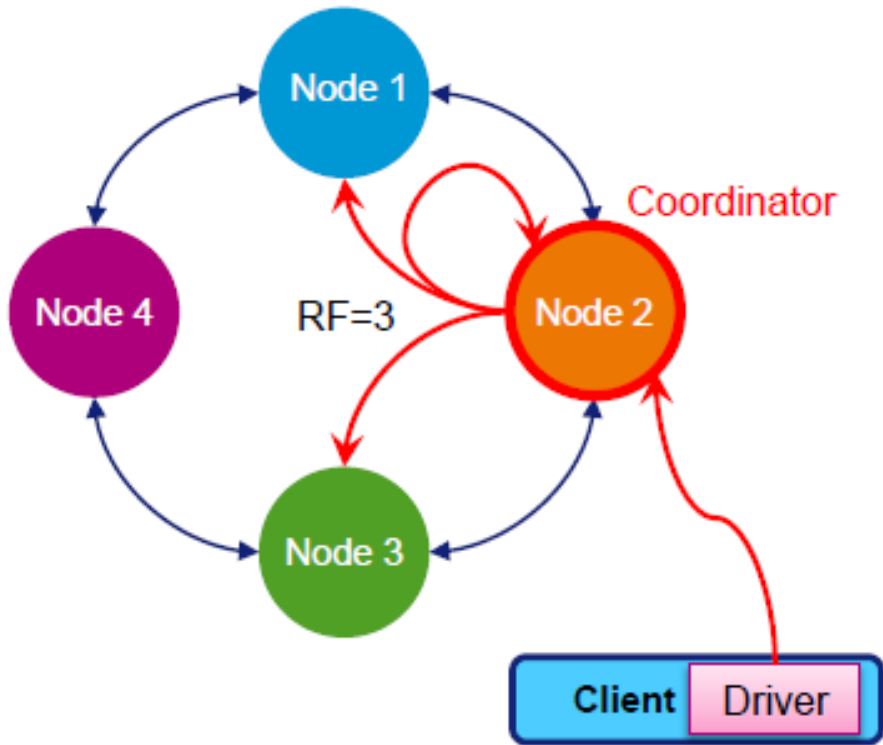


- Any node can coordinate any request
- Each client request may be coordinated by a different node

Replication of Data

- The **coordinator** manages the **replication process**
- **Replication Factor (RF)** : onto how many nodes should a write be copied
- The write will occur on the nodes responsible for that partition
- $1 \leq \mathbf{RF} \leq (\text{\#nodes in cluster})$
- Every write is time-stamped

Replication of Data



Transactions in Cassandra

- Cassandra is not a transactional database – no ACID.
- It uses “**AID**” - **A**tomic, **I**solated and **D**urable
- Cassandra offers **tunable** data **consistency** across a database cluster
- Developer or administrator can decide exactly how strong (*e.g., all nodes must respond*) or eventual (*e.g., just one node responds, with others being updated eventually*) they want data consistency to be.

Consistency : Quorum Algorithm

- The coordinator applies the Consistency Level (CL)
- **Consistency Level (CL)**: Number of nodes which must acknowledge a request
- **Examples of CL** : ONE , TWO , THREE , ANY , ALL (Not recommended)
- **QUORUM** = $(RF/2 + 1)$
- $CL = QUORUM$
- CL may vary for each request
- On success, the coordinator notifies the client

Cassandra : Application use cases

- Real-time, big data workloads
- Time series data management
- High-velocity device data consumption and analysis
- Media streaming management (e.g., music, movies)
- Social media (i.e., unstructured data) input and analysis
- Online web retail (e.g., shopping carts, user transactions)
- Real-time data analytics
- Online gaming (e.g., real-time messaging)
- Software as a Service (SaaS) applications that utilize web services
- Online portals (e.g., healthcare provider/patient interactions)
- Most write-intensive systems

Time series database example

- Cassandra is an excellent fit for time series data.
- Examples of TSDB use cases:
 - Ratings, recent purchases, and shopping cart
 - Session data, event streams, and click streams
 - Sensor data, application, and performance metrics
 - Velocities or windowed queries for a specific time period

Installation : Cassandra Clustering

- Install Cassandra on each node.
- Choose a name for the cluster.
- Get the IP address of each node.
- For each node, do the following configuration
- Go to **conf** folder in the **apache-cassandra** home directory.
- Open **cassandra.yaml** file in notepad and edit
- **listen_address**: <ip address of node>
- **rpc_address**: <ip address of node>
- - **seeds**: "<comma-delimited list of the IP address of each node in the cluster>"

Installation ...

- The communication of Cassandra nodes mainly revolves around 2 ports which are :
 - I. 7000 – TCP port for commands and data.
 - II. 9042 – TCP port for the native transport server *cqlsh*.
- Configure these ports on each node in the cluster
- Use *nodetool status* to see the status of cluster

```
anmols@anmols-15-3568:/opt/Applications/apache-cassandra-3.10$ ./bin/nodetool status
Datacenter: datacenter1
=====
Status=Up/Down
|| State=Normal/Leaving/Joining/Moving
-- Address            Load            Tokens      Owns (effective)  Host ID                               Rack
UN 192.168.2.122      7.28 MiB        256         49.4%             19fe4145-d49b-4009-b5d2-93c205452da0 rack1
UN 192.168.2.101     529.85 KiB      256         50.6%             74f9b27e-be87-411b-bae2-99ae1a657d7d rack1
```

- Cluster is now ready to use.

DataStax OpsCenter

- DataStax OpsCenter is a visual management and monitoring solution for Apache Cassandra and DataStax Enterprise.
- It simplifies administration tasks such as:
 - Adding and expanding clusters
 - Configuring nodes
 - Viewing performance metrics
 - Rectifying issues
 - Monitoring the health of your clusters on the dashboard
- Available as open source Cassandra and DataStax Enterprise.

OpsCenter architecture overview

The agents use Java Management Extensions (JMX) to monitor and manage each node.

