VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Cassandra Architecture

Lecturer* : *Dr. Pavle Mogin

SWEN 432
Advanced Database Design and Implementation

Cassandra The Fortune Teller

Plan for Cassandra Architecture

- Prologue
- Internode Communication
- Data Distribution and Replication
- Partitioners
- Snitches
- `CREATE KEYSPACE` command

Prologue

- Cassandra documentation describes the architecture in terms of:
  - Data centres,
  - Clusters,
  - Internode communication,
  - Data distribution and replication,
  - Snitches, and
  - Keyspaces
- Important structural information is held in two configuration files:
  - `cassandra.yaml`, and
  - `cassandra-topology.properties`

Data Centre and Cluster

- Data Centre
  - A collection of related nodes
  - Data centre is synonymous with replication group, that is, a grouping of nodes configured together for replication purposes
  - A data centre can be physical or virtual
  - If a cluster spans more then one data centre
    - Replication is set by data centre
    - The same data is written in all data centre
  - Using separate data centres allows:
    - Dedicating each data centre for different processing tasks
    - Satisfying requests from a datacentre close to client, and
    - Improving availability for the same level of consistency
    - Data centres should never span physical locations
- Cluster
  - A cluster contains nodes belonging to one or more data centres
  - It can span physical locations

Internode Communication

- Cassandra uses Gossip communication protocol in which nodes periodically exchange state information about themselves and other nodes they know about
- The gossip process runs every second and exchanges messages with up to three other nodes in the cluster
- Gossip messages are versioned
  - During a gossip exchange, older information is overwritten with the most current state for a particular node

Configuring Gossip Settings and Seeds

- When a node first starts up, it looks at its `cassandra.yaml` configuration file to determine:
  - The name of the Cassandra cluster it belongs to,
  - Which nodes (called *seeds*) to contact to obtain information about the other nodes in the cluster, and
  - Other parameters for determining port and range information
- All these parameters are set by an administrator and have to be the same for all nodes of a cluster
- The property `seed_provider` is a list of comma-delimited hosts (IP addresses)
- In multiple data-center clusters, the seeds list should include at least one node from each data centre

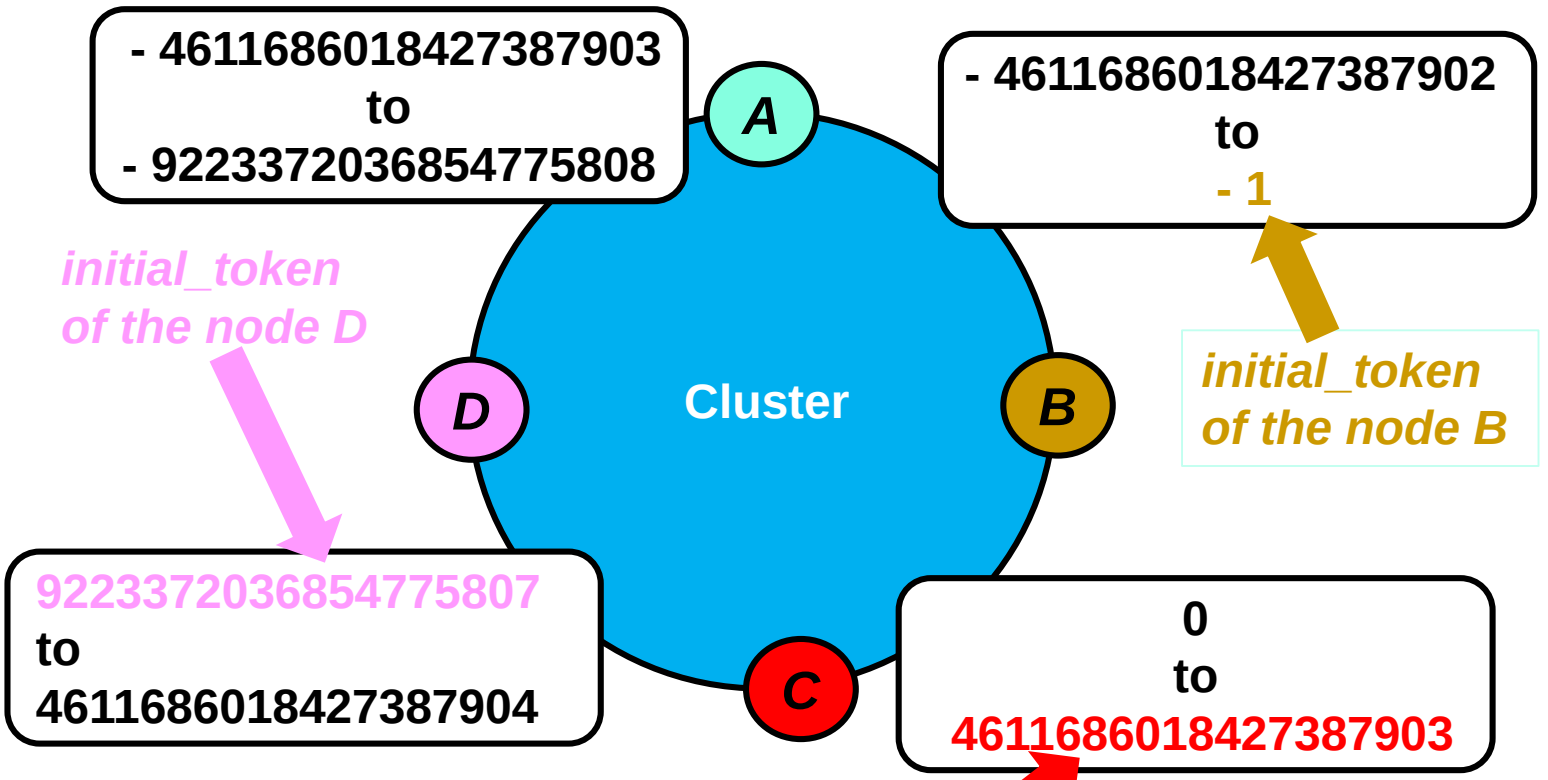
Node Failure – Dynamic Snitch

- During gossiping, every node maintains a sliding window of inter-arrival times of gossip messages from other nodes in the cluster
  - That mechanism is called **dynamic snitch**
- If a node does not respond during an adjustable time period, it is either down or experiencing transient problems
- Cassandra uses this information to avoid routing client requests to unreachable nodes (if possible), or nodes that are performing poorly
- The live nodes periodically try to re-establish contact with failed nodes to see if they are back up

Data Distribution and Partitioning

- Data distribution is done by Consistent Hashing
- Partitioning
  - A partitioner determines how data is distributed across the nodes in the cluster
  - Basically, a partitioner is a function for deriving a token representing a row from its partition key
  - Cassandra default partitioner is a hash function named `Murmur3Partitioner`
  - This hashing function creates a 64-bit value of the partition key
  - The possible range of hash values is from -2^{63} to $+2^{63}-1$.

Hashing Values in a Cluster



$2^{63} - 1 = 9223372036854775807$

$-2^{63} = -9223372036854775808$

Only physical nodes assumed

Virtual Nodes

- Virtual nodes are used to:
  - Balance the work load.
  - Add or remove nodes in an easy way, and
  - Rebuild a dead node faster
- For each physical node of a cluster, the administrator specifies the number of virtual nodes in its `cassandra.yaml` configuration file
- The property `num_tokens` defines the number of tokens randomly assigned to this node on the ring
  - One token for each virtual node

Data Replication

- Cassandra supports a number of replication strategies:
  - Simple Strategy,
  - Network Topology Strategy,
  - Gossiping Network Topology Strategy,
  - Amazon Worldwide Services (EC2 and EC2 Multi Region)
- We consider the Simple Strategy and Network Topology Strategy, only
- Simple Strategy is used for single data centres only
  - Places the first replica on a node determined by the partitioner
  - Additional replicas are placed on the next nodes clockwise in the ring without considering topology (rack or data centre location)

Network Topology Strategy

- Network Topology Strategy (NTS) should be used for clusters deployed across multiple data centres
- NTS is:
  - Data Centre aware and
  - Rack aware
- This strategy specifies the number of replicas in each data centre
- NTS places replicas in the same data centre by walking the ring clockwise until reaching the first node in another rack
  - NTS attempts to place replicas on distinct racks because nodes in the same rack (or similar physical grouping) often fail at the same time due to power, cooling, or network issues

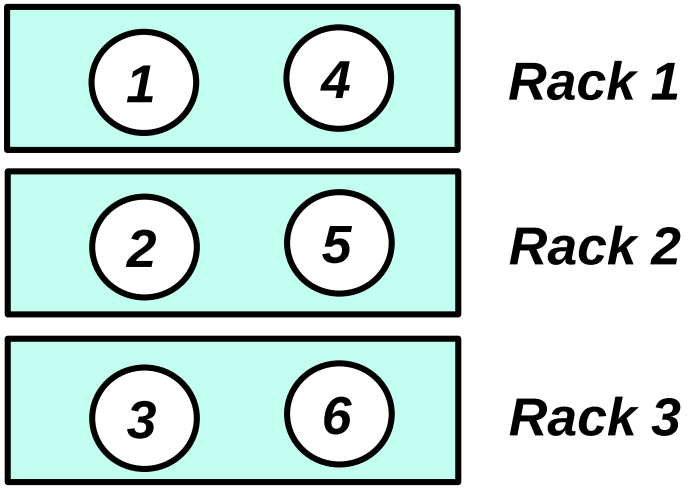
Configuring Data Centre Replication

- Primary considerations when configuring replicas in a multi data centre cluster are:
  - Being able to satisfy reads locally, without incurring cross data-centre latency, and
  - Failure scenarios
- Failure scenarios:
  - Two replicas in each data centre tolerates the failure of a single node per replication group and still allows local reads at a consistency level of ONE.
  - Three replicas in each data centre tolerates either the failure of one node per replication group at a strong consistency level of LOCAL_QUORUM or multiple node failures per data centre using consistency level ONE
  - Asymmetrical replication groupings are also possible:
    - Three replicas in one data centre for real-time applications, and
    - A single replica in the other for running analytics

Example 1

- Assume:
  - A six node cluster has been deployed using a single data centre
  - The availability requirements are:
    - Strong consistency for 100% of data when one node is down
    - Strong consistency for 100% of data when one rack is down
- How many racks the cluster should be deployed to and how many nodes per a rack to achieve availability requirements? What should be the replication factor?
- Answer:

$n = 3$

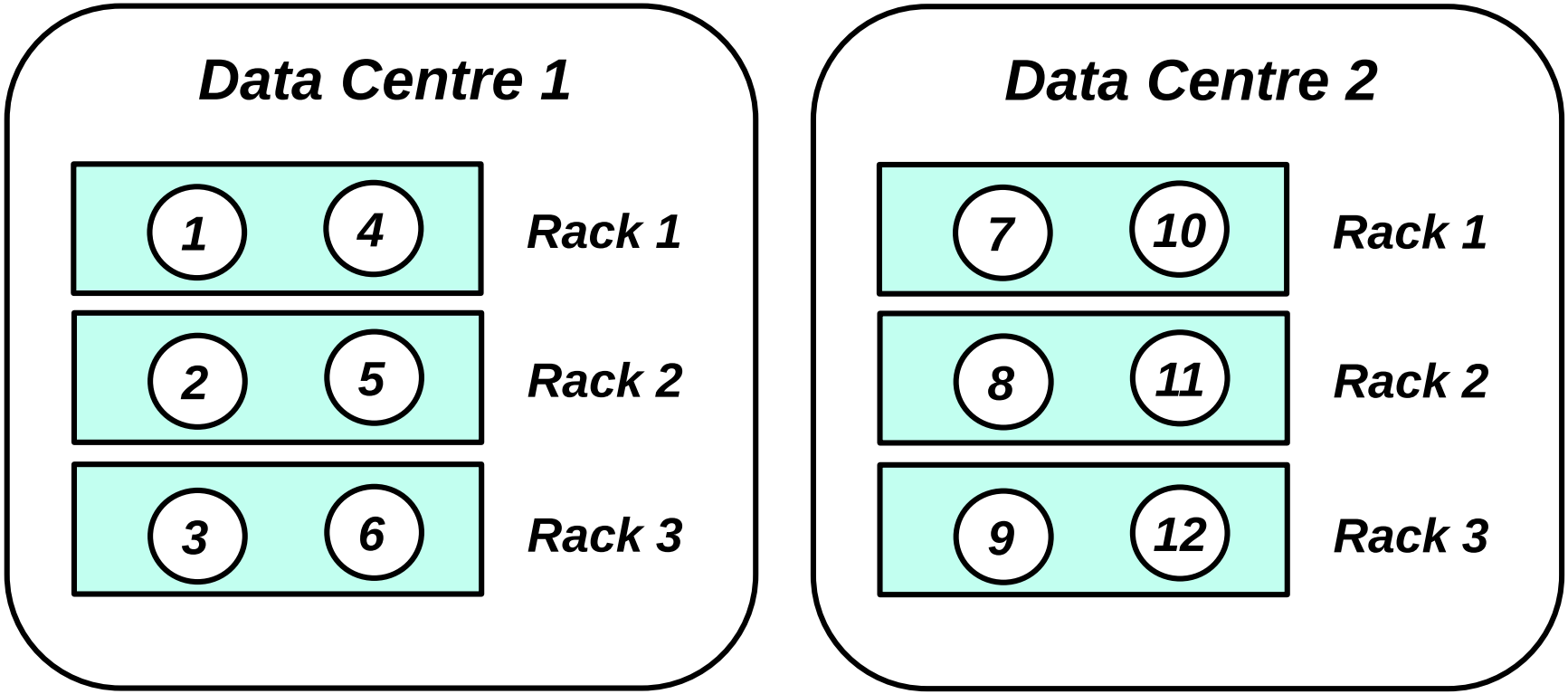


Example 2

- Assume:
  - A twelve node cluster has been deployed using two data centres, containing six nodes each
  - Each physical node has only one virtual node
  - The availability requirements are:
    - Strong consistency for 100% of data when two nodes are down
    - Strong consistency for 100% of data when two racks are down
- How many racks the cluster should be deployed to and how many nodes per a rack to achieve availability requirements?
- What should be the replication factor?

Answer

Local replication factors $n_1 = 3$ and $n_2 = 3$
Global replication factor $n = 6$



Local quorums $q_1 = 2$ and $q_2 = 2$
Global quorum $q = 4$

Snitches

- A snitch determines which data centres and racks nodes belong to
- They inform Cassandra about the network topology so that requests are routed efficiently and allows Cassandra to distribute replicas by grouping machines into data centres and racks
- Cassandra supports:
  - SimpleSnitch,
  - PropertyFileSnitch,
  - Others
- Simple Snitch:
  - The SimpleSnitch is the default snitch
  - It is used for single-data centre deployments, only
  - It does not recognize data centre or rack information

Property File Snitch

- This snitch determines proximity by rack and data centre
- It uses the network details located in the `cassandra-topology.properties` file
- When using this snitch, the administrator can define data centre names to be whatever he/she wants
- Data centre names have to correlate to the name of data centres in the keyspace definition
- Every node in the cluster should be described in the `cassandra-topology.properties` file, and this file should be exactly the same on every node in the cluster

CREATE KEYSPACE

- The replication strategy and the replication configuration are defined within the `CREATE KEYSPACE` CQL command

```
CREATE KEYSPACE
IF NOT EXISTS keyspace_name
WITH REPLICATION = map AND
DURABLE_WRITES = ( true | false )
```

- The `map` is used to declare:
  - replica placement strategy class (either Simple or Network Topology) , and
  - Replication configuration (factor)

The map of CREATE KEYSPACE

- The two different types of keyspaces:

```
{ 'class' : 'SimpleStrategy',
  'replication_factor' : <integer> };
```

```
{ 'class' : 'NetworkTopologyStrategy'
[, '<data center>' : <integer>,
'<data center>' : <integer>] . . . };
```

- The `SimpleStrategy` is used for evaluating and testing Cassandra
  - It is correlated with the `SimpleSnitch`
- The `NetworkTopologyStrategy` should be used for production or for use with mixed workloads
  - It is correlated with `PropertyFileSnitch` that uses information in `cassandra-topology.properties` file

Configuring NTS

- Before creating a `keyspace` for use with multiple data centres, the cluster has to be configured to use a network-aware snitch
  - The `cassanra.yaml` configuration file of each node has to be configured before starting the cluster
  - One of settings to be checked and (possibly) done is

    `endpoint_snitch: PropertyFileSnitch`
- For a cluster spanning multiple data centres, data centre names and rack information have to be checked and possibly changed in `cassandra-topology.properties` files of each node
  - Cassandra uses `dci, i = 1,…,` as the default data centre name
  - Centre names in `.properties` file and in the `map` of `CREATE KEYSPACE` have to match exactly, otherwise Cassandra will fail to find a node, to complete a write request

Setting Durable_Writes

- `DURABLE_WRITES` is an option whose default is `yes`
- When set to `false`, data written to the keyspace bypasses the commit log
  - A risk of losing data
  - Not to use on a key space using the `SimpleStrategy`

Summary (1)

- A data centre is a collection of nodes configured for replication purposes
- A cluster contains nodes from one or more data centres
- Internode communication is accomplished through gossiping
- The dynamic snitch is a mechanism for detecting poorly performing or failing nodes
- Data distribution is done by Consistent Hashing
  - Partitioning is performed by deriving a token from the partition key of a row and storing the row on a node assigned to the first greater token on the consistent hashing ring

Summary (2)

- Data replication is performed according to one of replication strategies:
  - Simple Strategy,
  - Network Topology Strategy
- A snitch contains information about the network topology
  - Each snitch corresponds to one replication strategy and vice versa
- The replication strategy and the replication configuration (factor) are declared within a keyspace declaration