
DATASTAX
KubeCon 2020 Workshop
Apache Cassandra and Kubernetes

Apache Cassandra™ with Kubernetes

1. **Housekeeping and Quiz**
2. Cassandra Basics
3. Kubernetes Operators
4. Cass Operator in Deep
5. Grafana | Prometheus
6. Resources

DATASTAX

DATASTAX

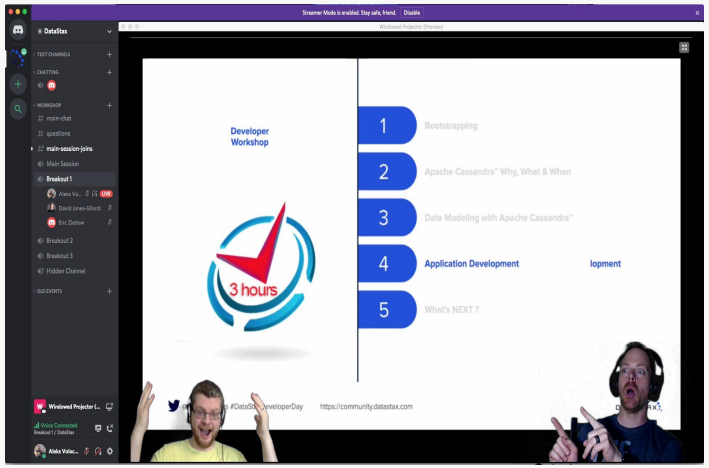
Disclaimer

Today,

we won't introduce the basics of Docker and Kubernetes

It's a KubeCon in the end!

STREAMS

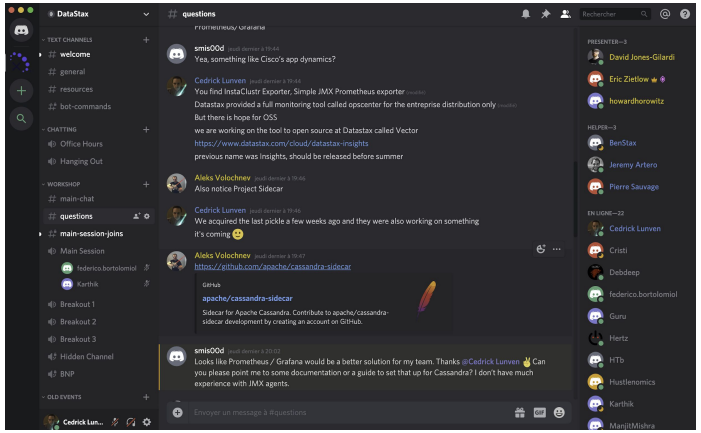


YouTube

zoom

RUNTIME

QUESTIONS



Discord

zoom

MATERIALS

Slides

Notebooks

DockerFile

Github

YOUR LAPTOP

CLOUD INSTANCE

http://kubecon2020.datastaxtraining.com/

Your Laptop

docker

kubernetes

kind

>_

Cloud Env

kubernetes

docker

kind

CentOS

aws

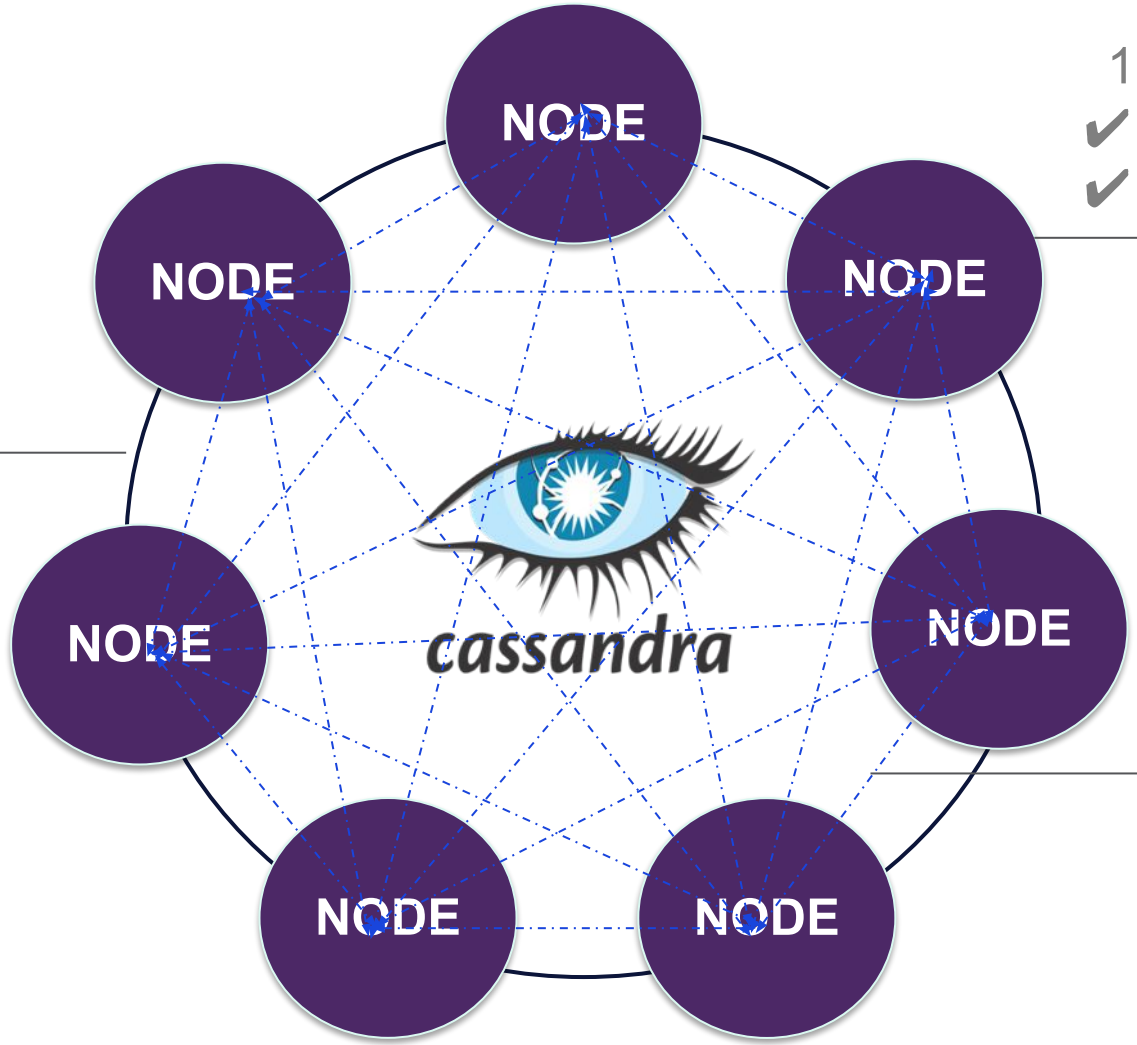

menti.com
51 63 695
Mentimeter
What is the most desirable feature of your ideal workplace?
Flexible time
Social factor
Big paycheck
Available on the iPhone
App Store
GET IT ON
Google play

Apache Cassandra™ with Kubernetes

1. Housekeeping and Quizz
2. Cassandra Basics
3. Kubernetes Operators
4. Cass Operator in Deep
5. Grafana | Prometheus
6. Resources

DATASTAX

Apache Cassandra™ = NoSQL Distributed Database



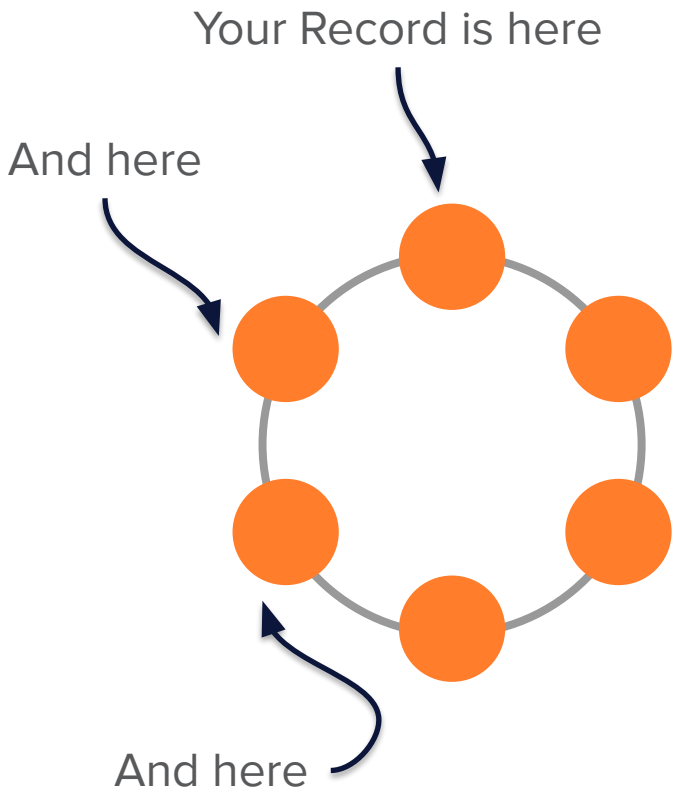
1 Installation = 1 NODE
- ✔ Capacity: ± 1TB
- ✔ Throughput: 3000 Tx/sec/core

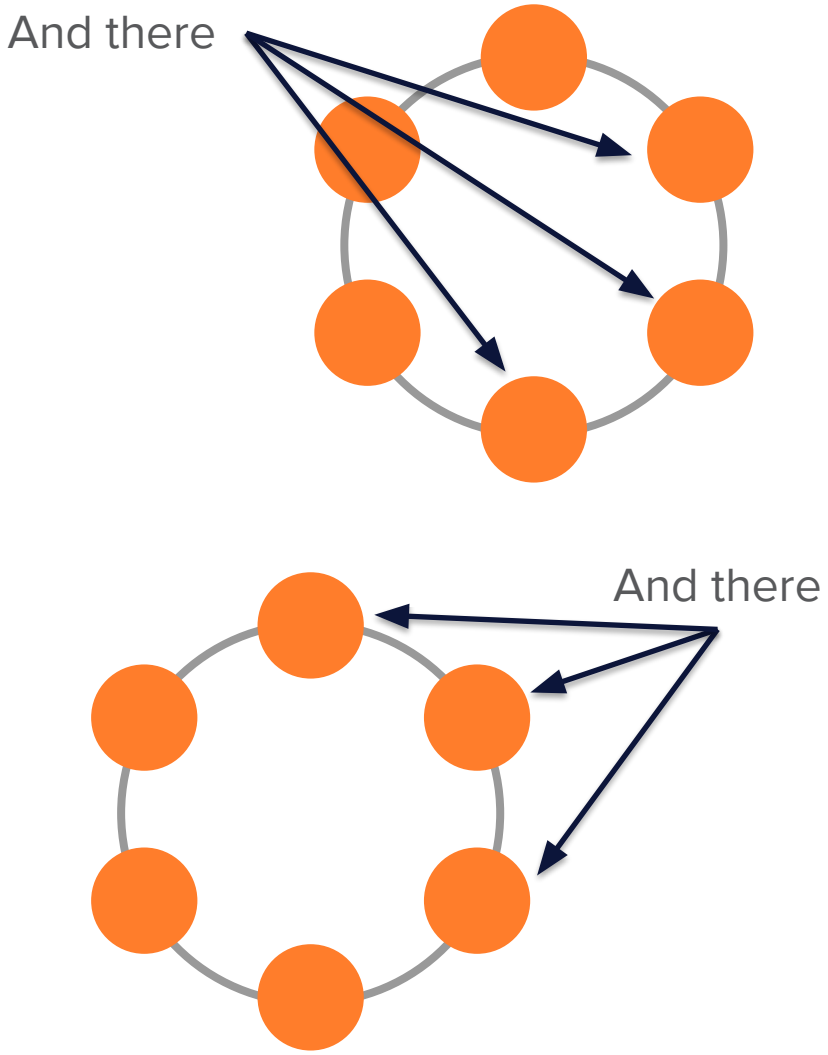
DataCenter | Ring

Communication:
- ✔ Gossiping

Cassandra Features

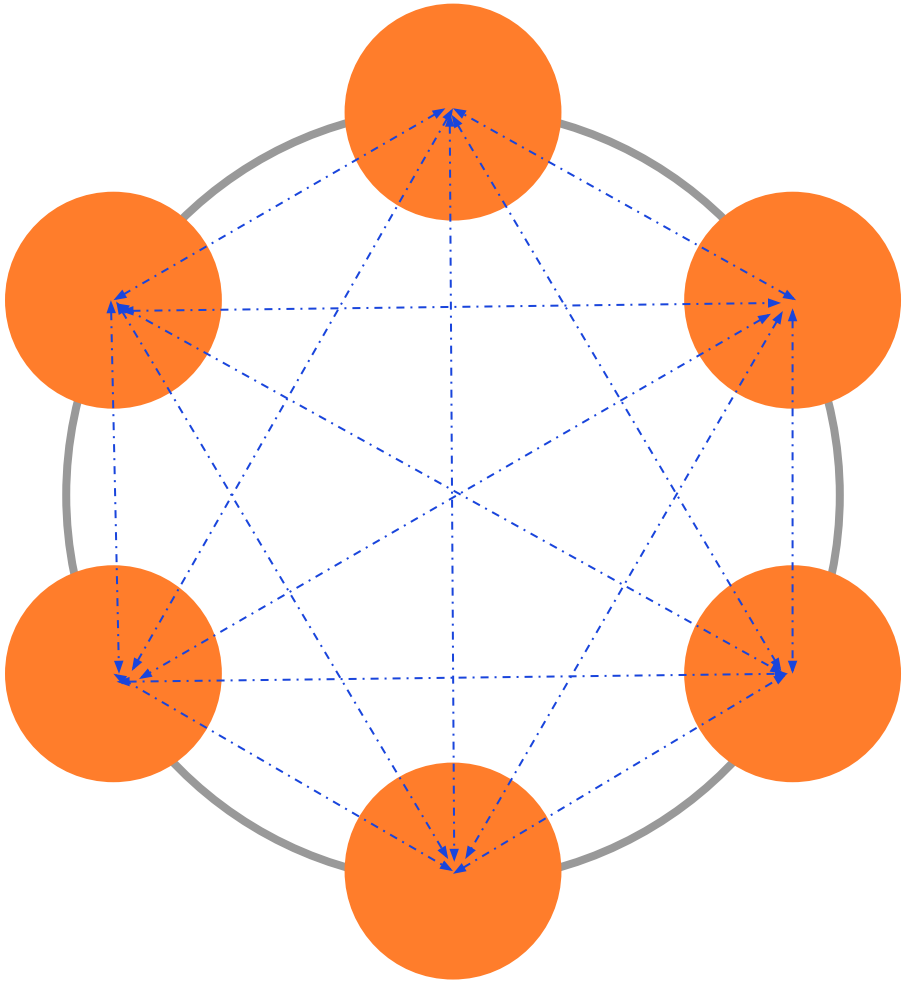
- Distributed
  - Responsive
  - Scalable
- Replicated
  - Available
- Masterless
  - No SPoF





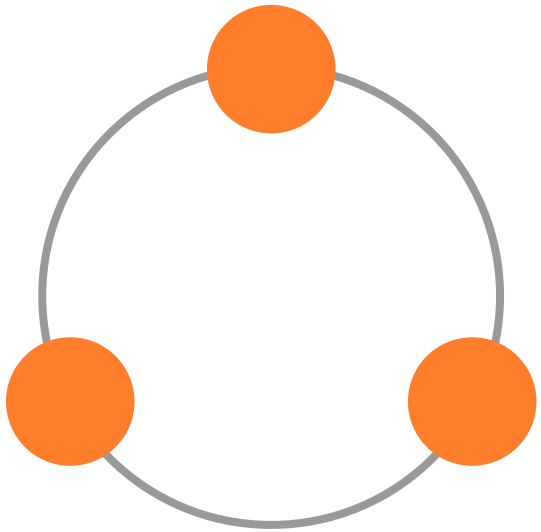
Cassandra is Distributed and Highly Available

- Nodes organized in a ring
- Nodes monitor each other's health status using a gossip protocol
- Cassandra routes requests away from nodes that are down or slow

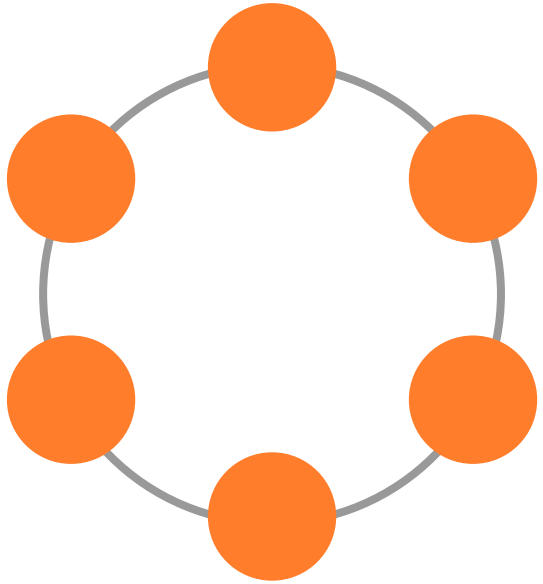


Horizontal vs. Vertical Scaling

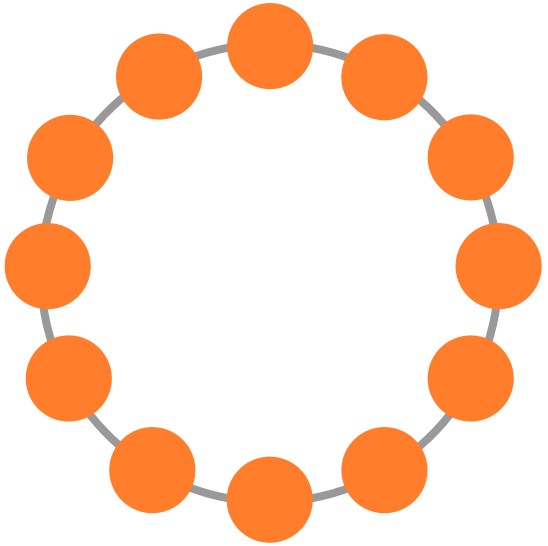
- Vertical scaling requires one large expensive machine
- Horizontal scaling requires multiple less-expensive commodity hardware



100,000 transactions/second



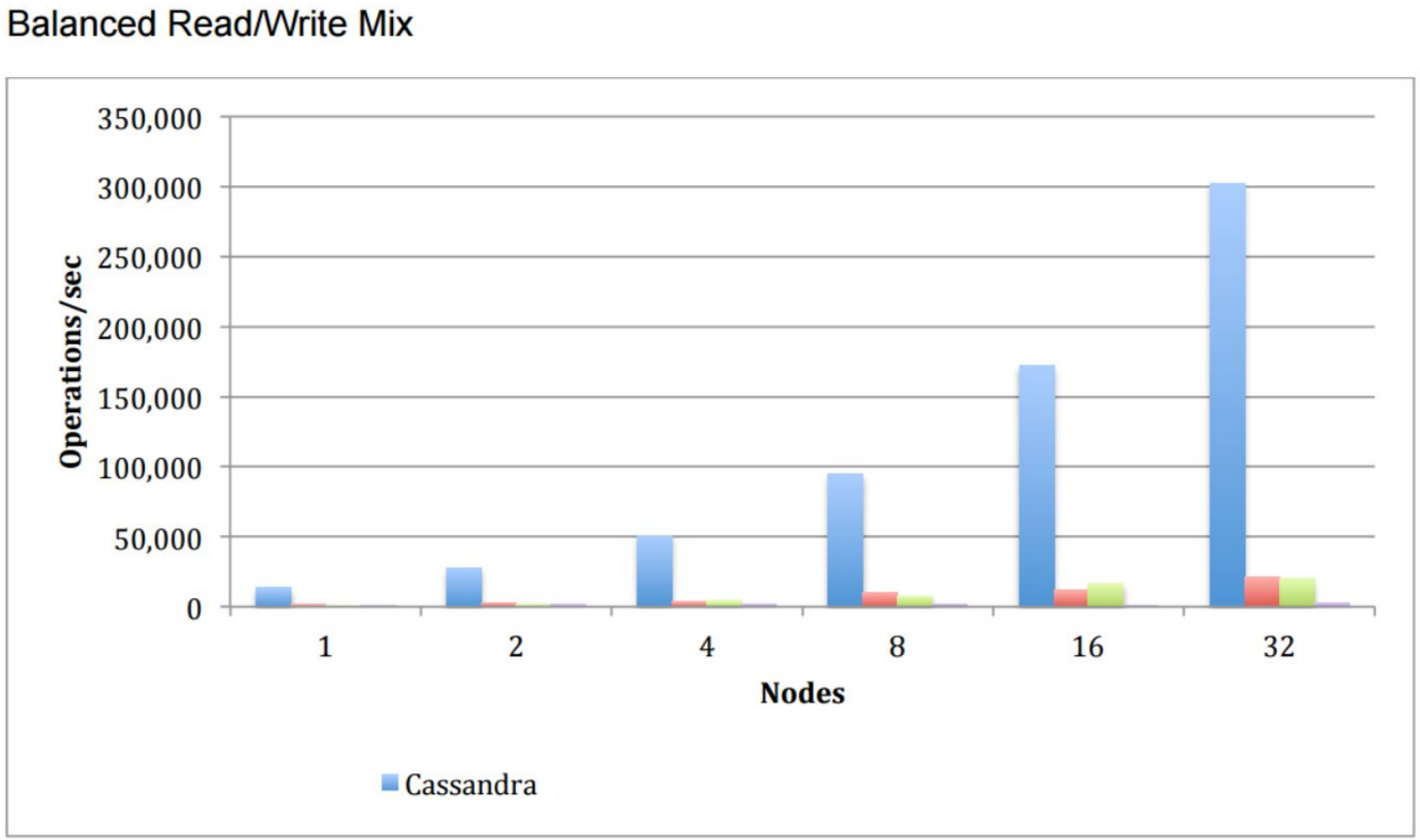
200,000 transactions/second



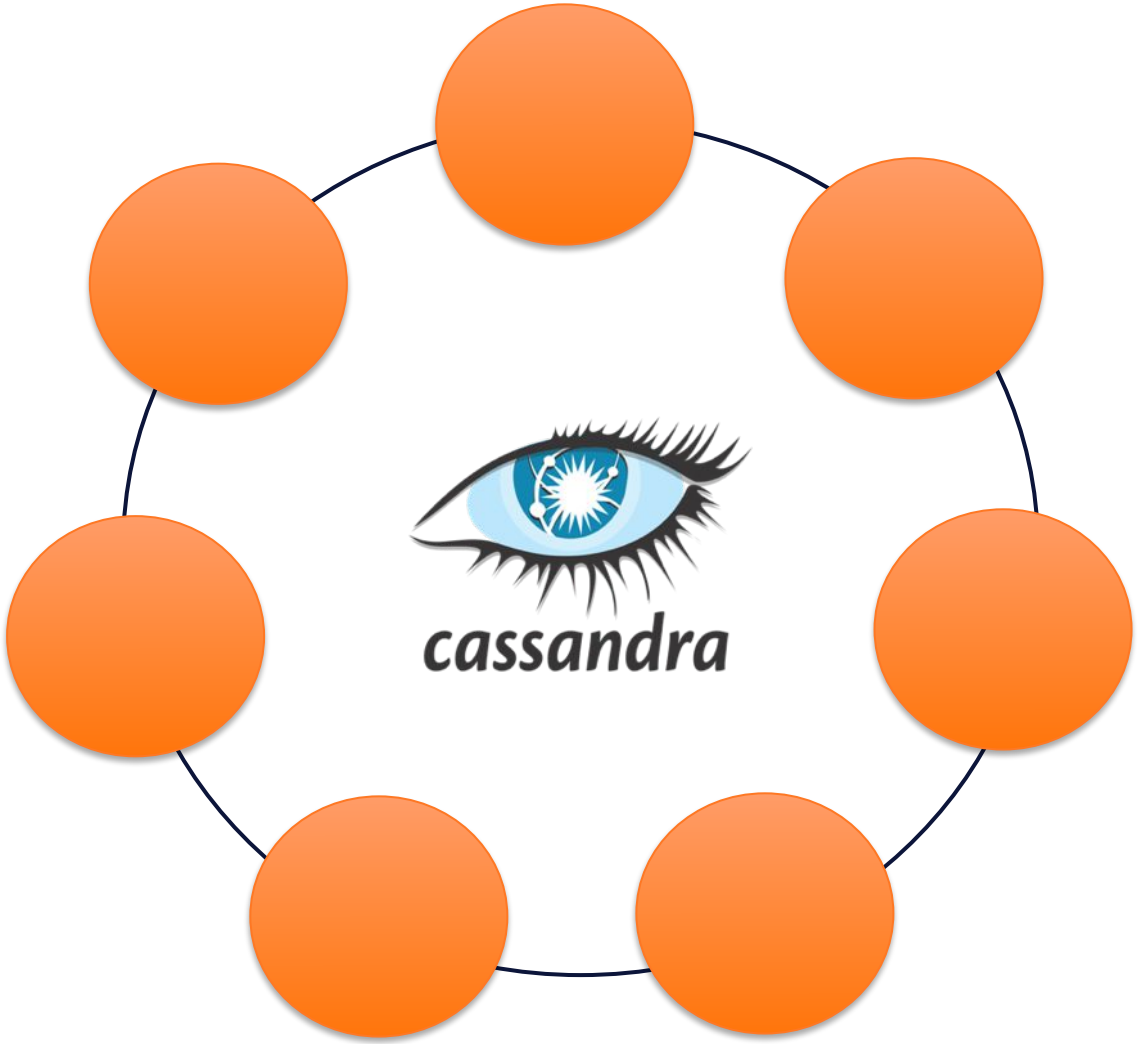
400,000 transactions/second

Scales Linearly

- Need more capacity?
- Need more throughput?
- Add nodes!



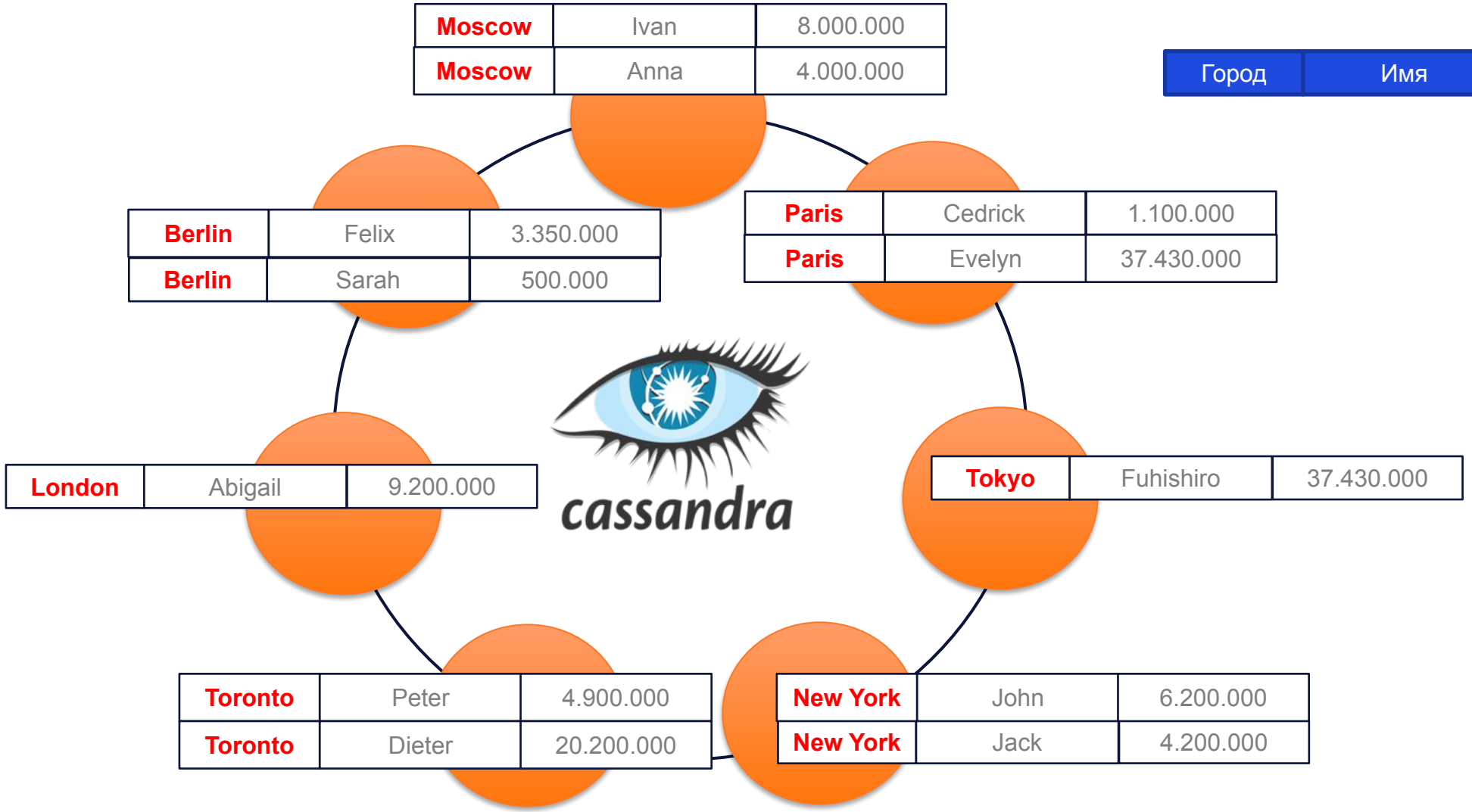
Data is Distributed?



Город	Имя	Телефон
Moscow	Ivan	8.000.000
Moscow	Anna	4.000.000
SPb	Aleksandr	2.230.000
Berlin	Felix	3.350.000
London	John	9.200.000
Sydney	Abigeil	4.900.000
Berlin	Sarah	500.000
Toronto	Andrew	6.200.000
Toronto	Peter	4.200.000
Paris	Cedrick	1.100.000
Paris	Evelyn	37.430.000
Mumbai	Prabhakharan	20.200.000

Partition Key

Data is Distributed



Город	Имя	Телефон

Data is Replicated

Город	Имя	Телефон

Moscow	Ivan	8.000.000
Moscow	Anna	4.000.000
Berlin	Felix	3.350.000
Berlin	Sarah	500.000

Berlin	Felix	3.350.000
Berlin	Sarah	500.000
London	Abigail	9.200.000

Paris	Cedrick	1.100.000
Paris	Evelyn	37.430.000
Moscow	Ivan	8.000.000
Moscow	Anna	4.000.000

London	Abigail	9.200.000
Toronto	Peter	4.900.000
Toronto	Dieter	20.200.000

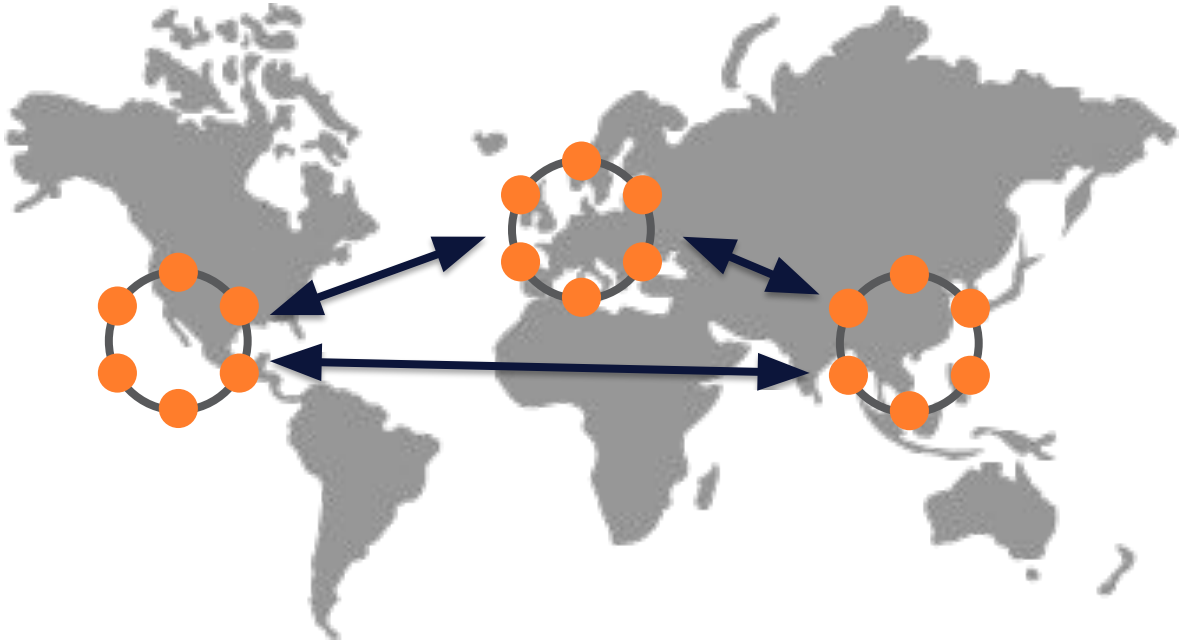
Tokyo	Fuhishiro	37.430.000
Paris	Cedrick	1.100.000
Paris	Evelyn	37.430.000

Toronto	Peter	4.900.000
Toronto	Dieter	20.200.000
New York	John	6.200.000
New York	Jack	4.200.000

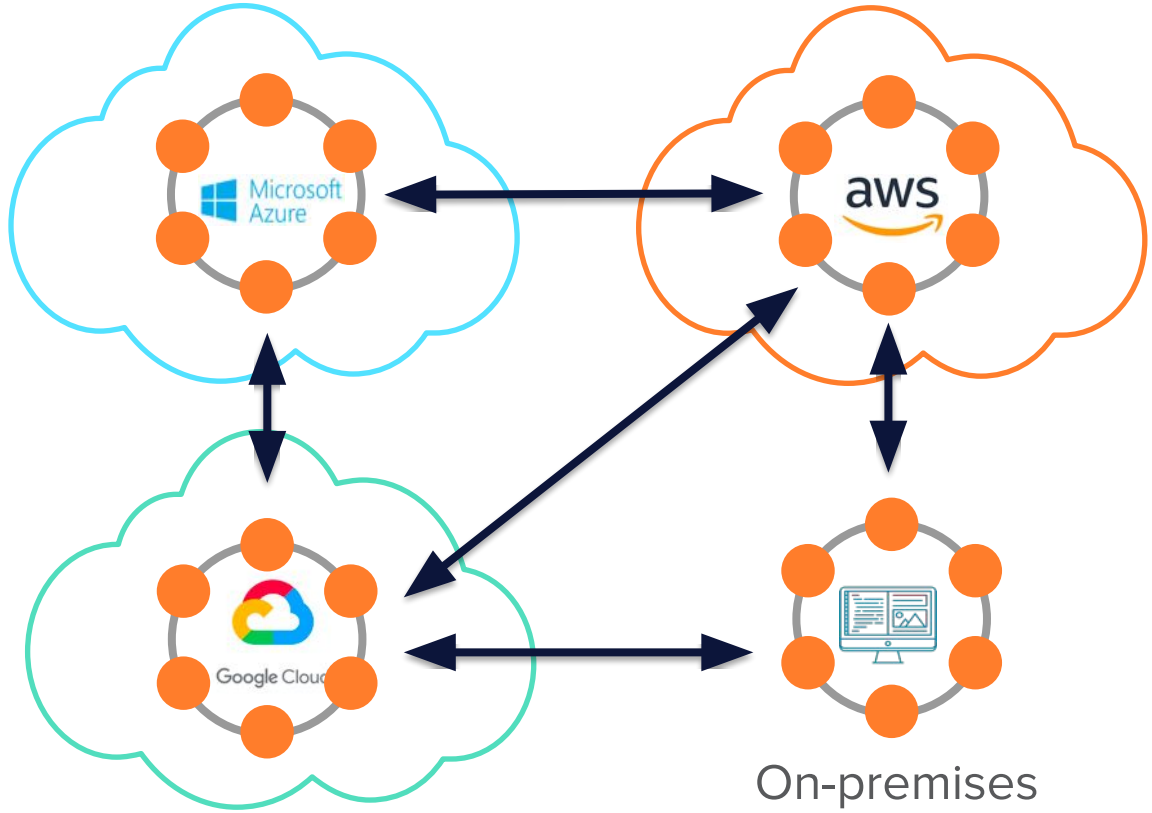
New York	John	6.200.000
New York	Jack	4.200.000
Tokyo	Fuhishiro	37.430.000

Data Distributed Everywhere

- Geographic Distribution



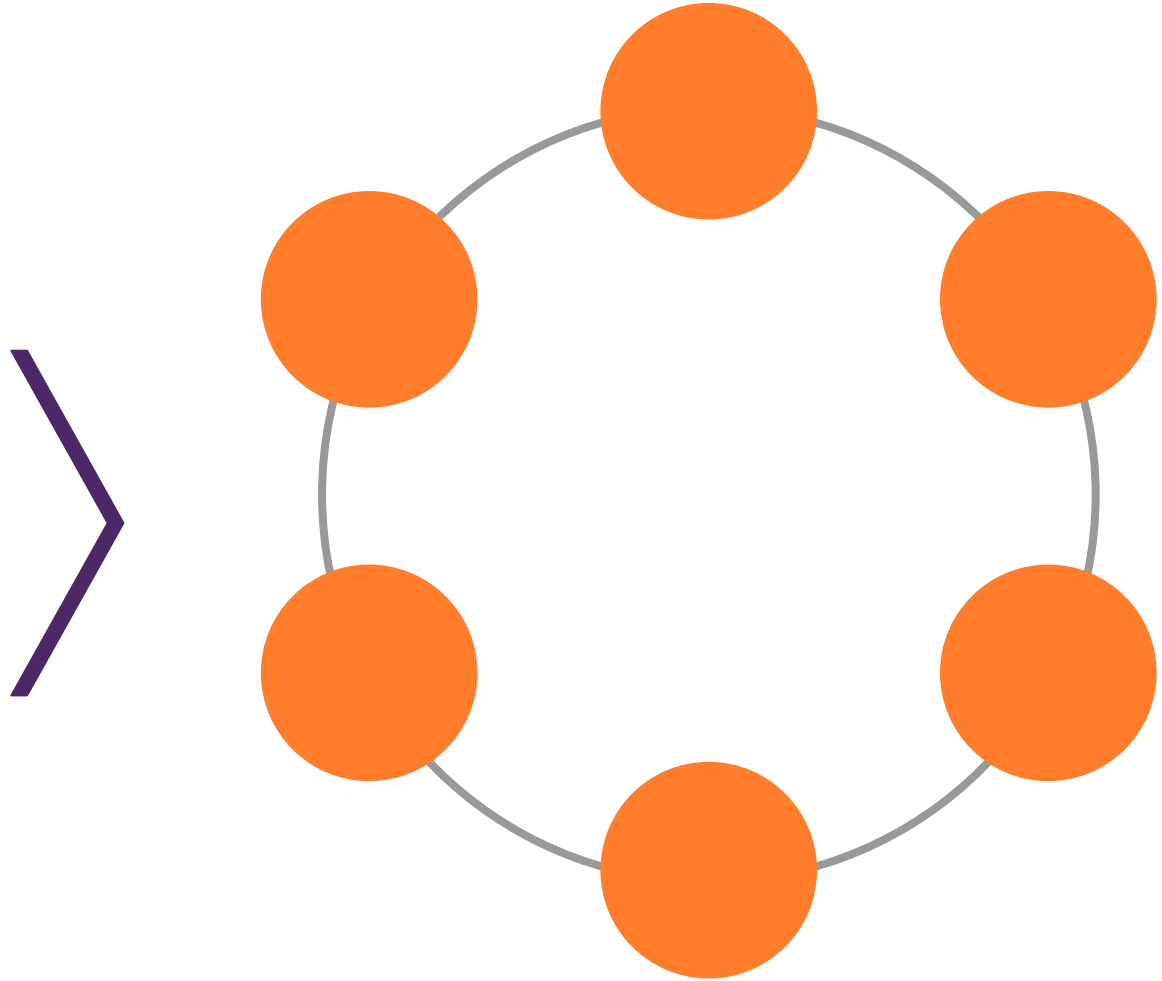
- Hybrid-Cloud and Multi-Cloud



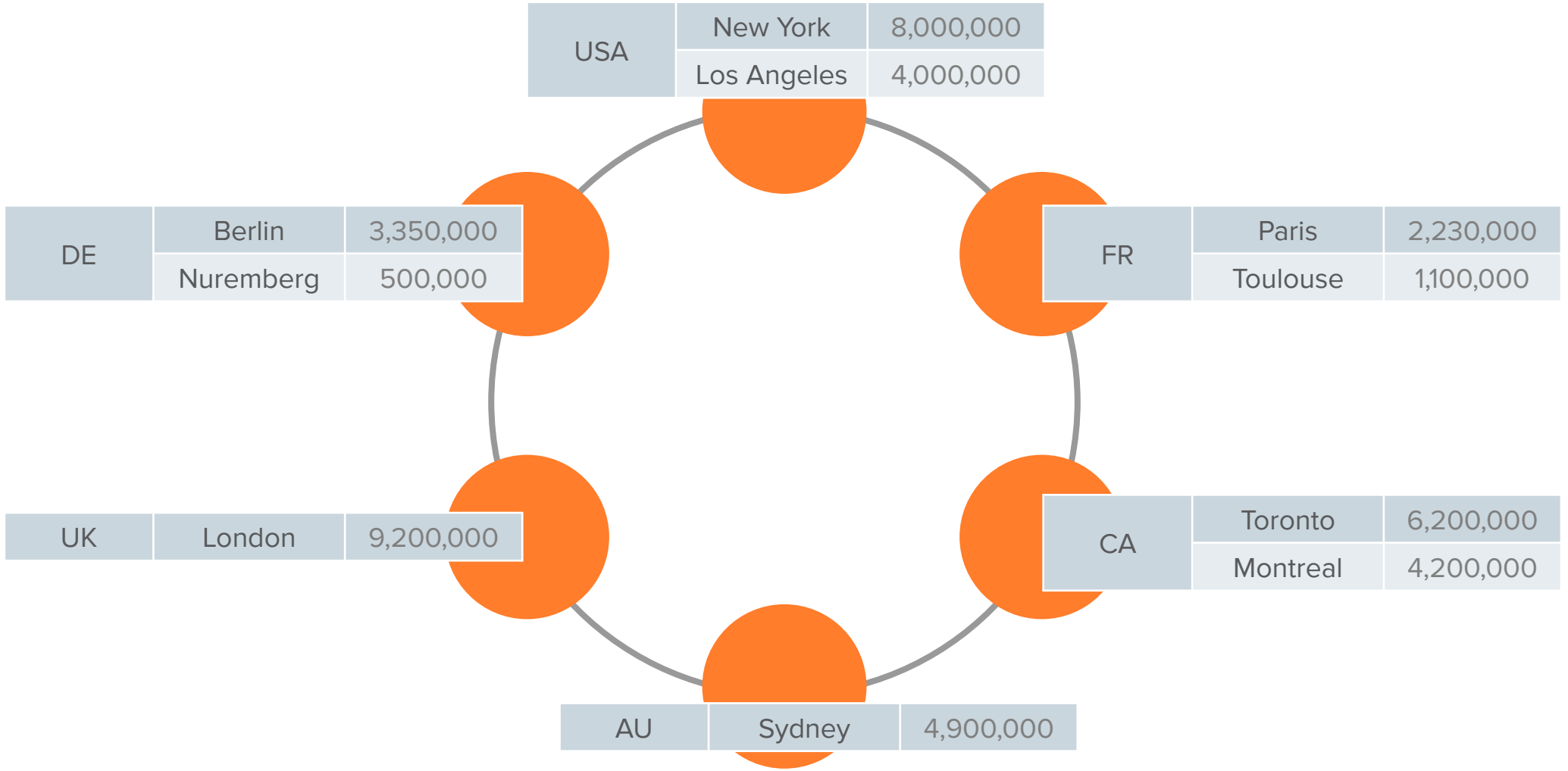
How the Ring Works

- How is the data distributed?

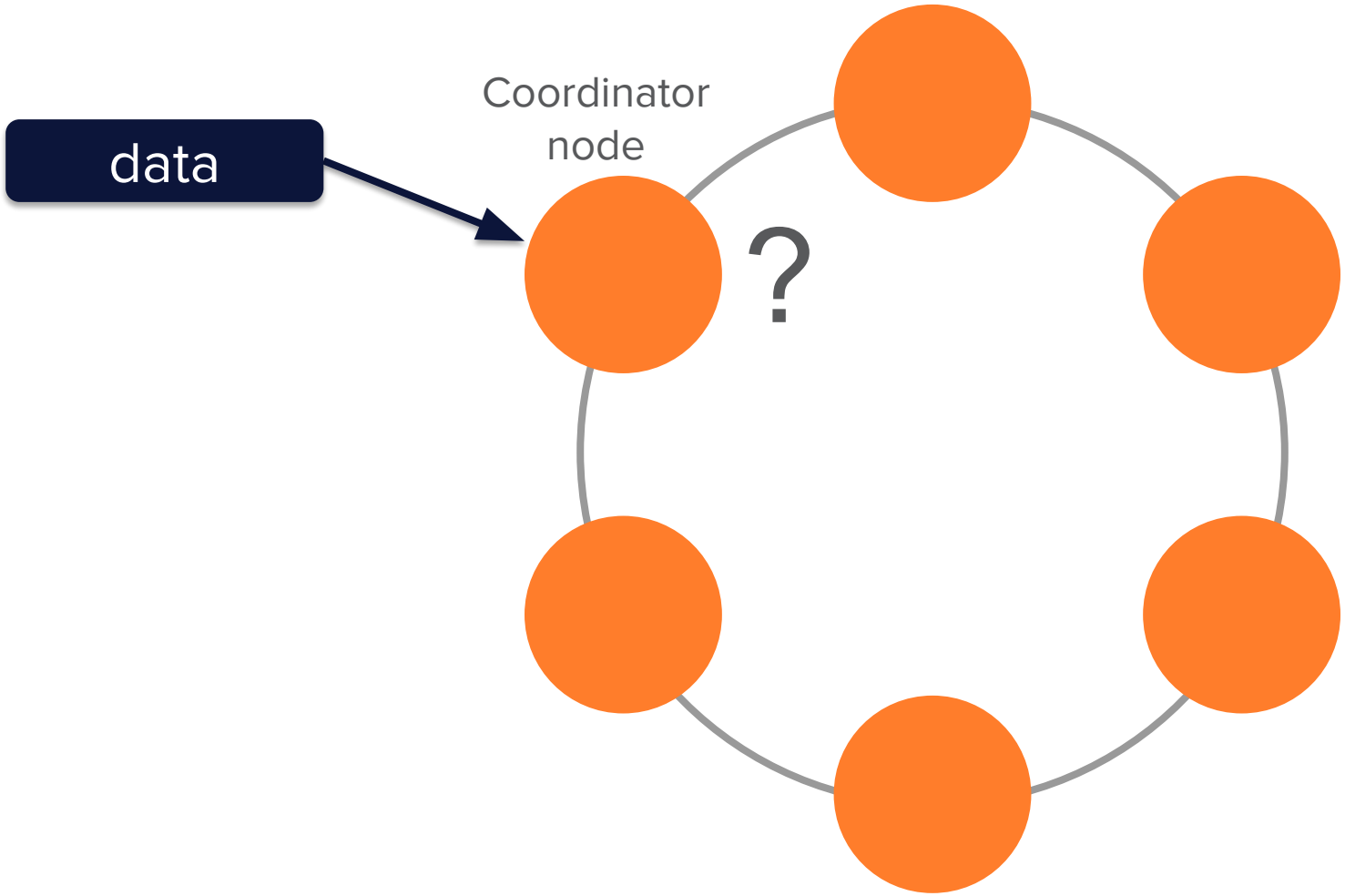
Country	City	Population
USA	New York	8,000,000
	Los Angeles	4,000,000
DE	Berlin	3,350,000
	Nuremberg	500,000
FR	Paris	2,230,000
	Toulouse	1,100,000
CA	Toronto	6,200,000
	Montreal	4,200,000
UK	London	9,200,000
AU	Sydney	4,900,000



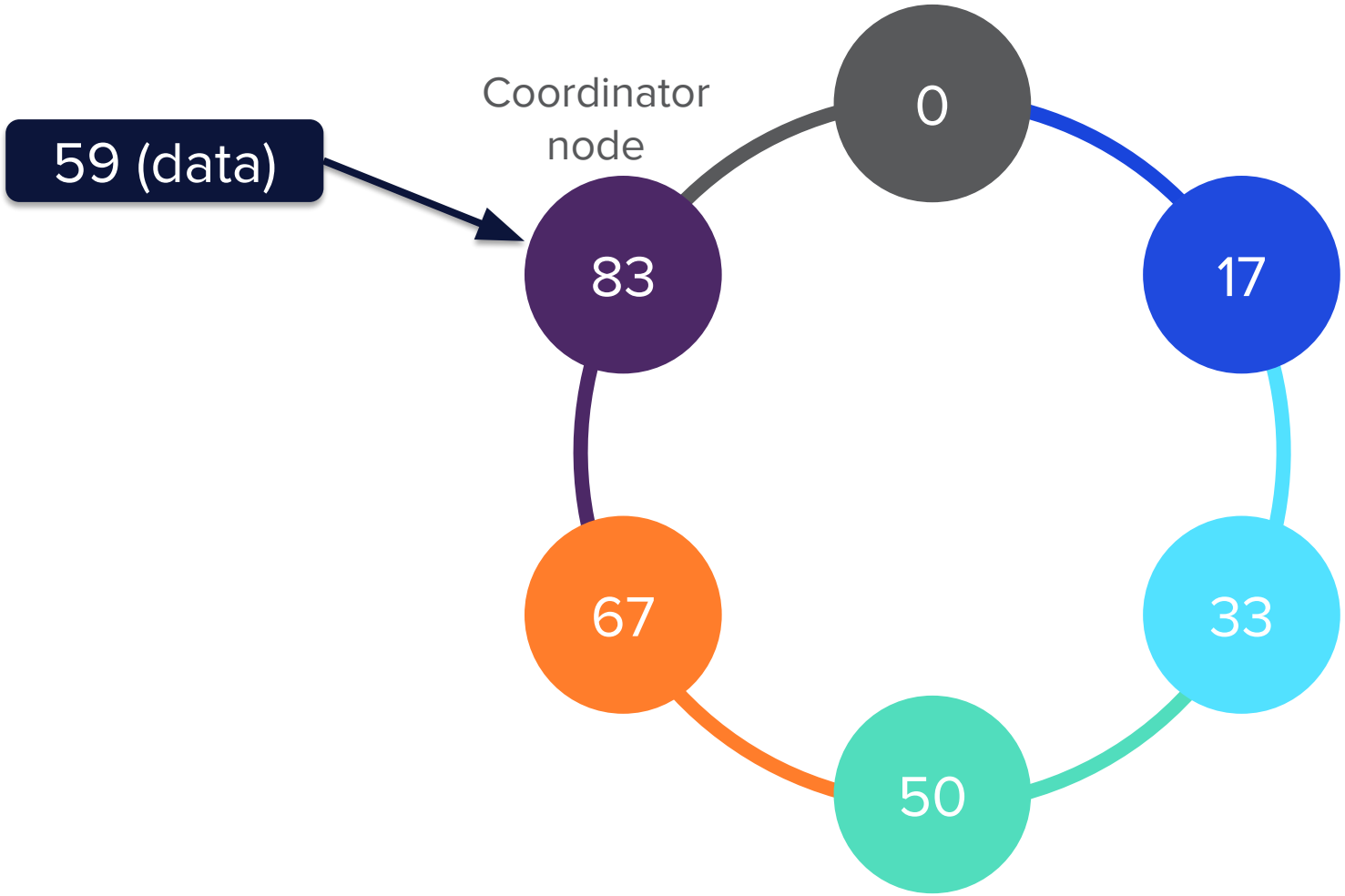
How the Ring Works



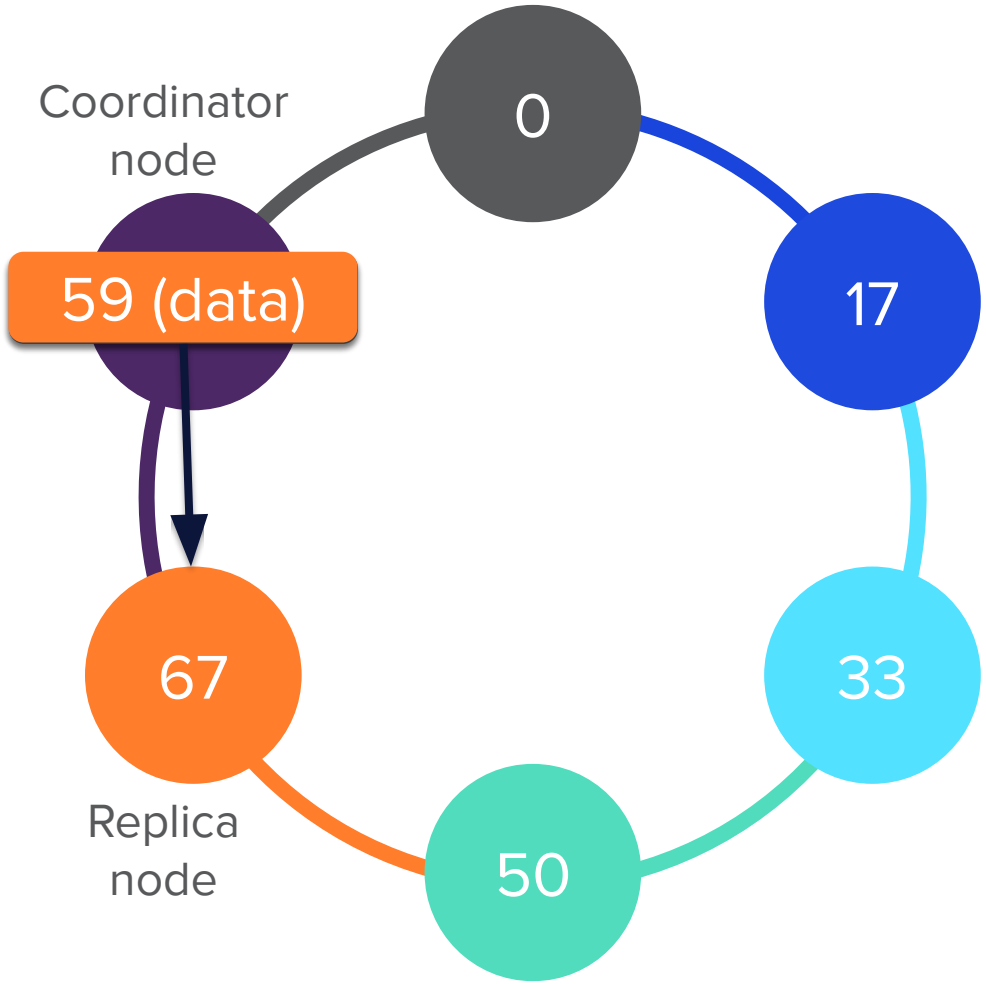
How the Ring Works



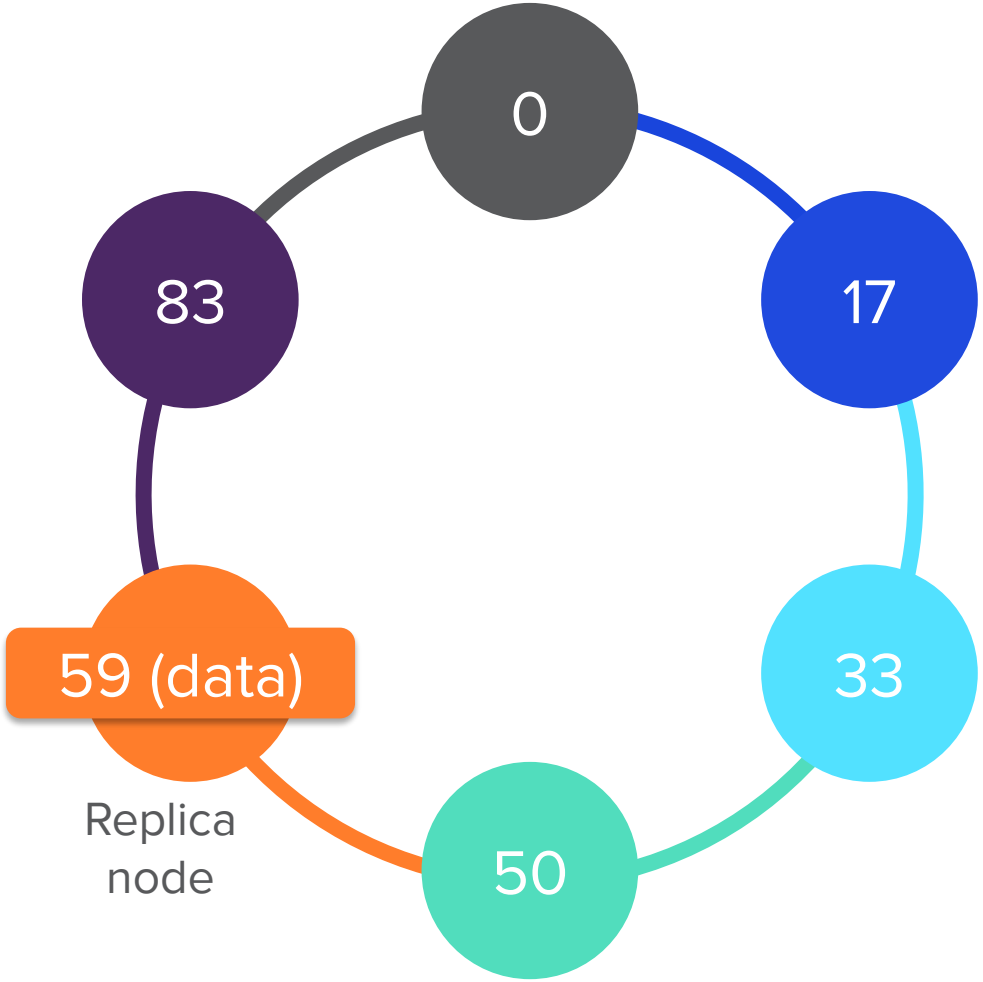
How the Ring Works



How the Ring Works

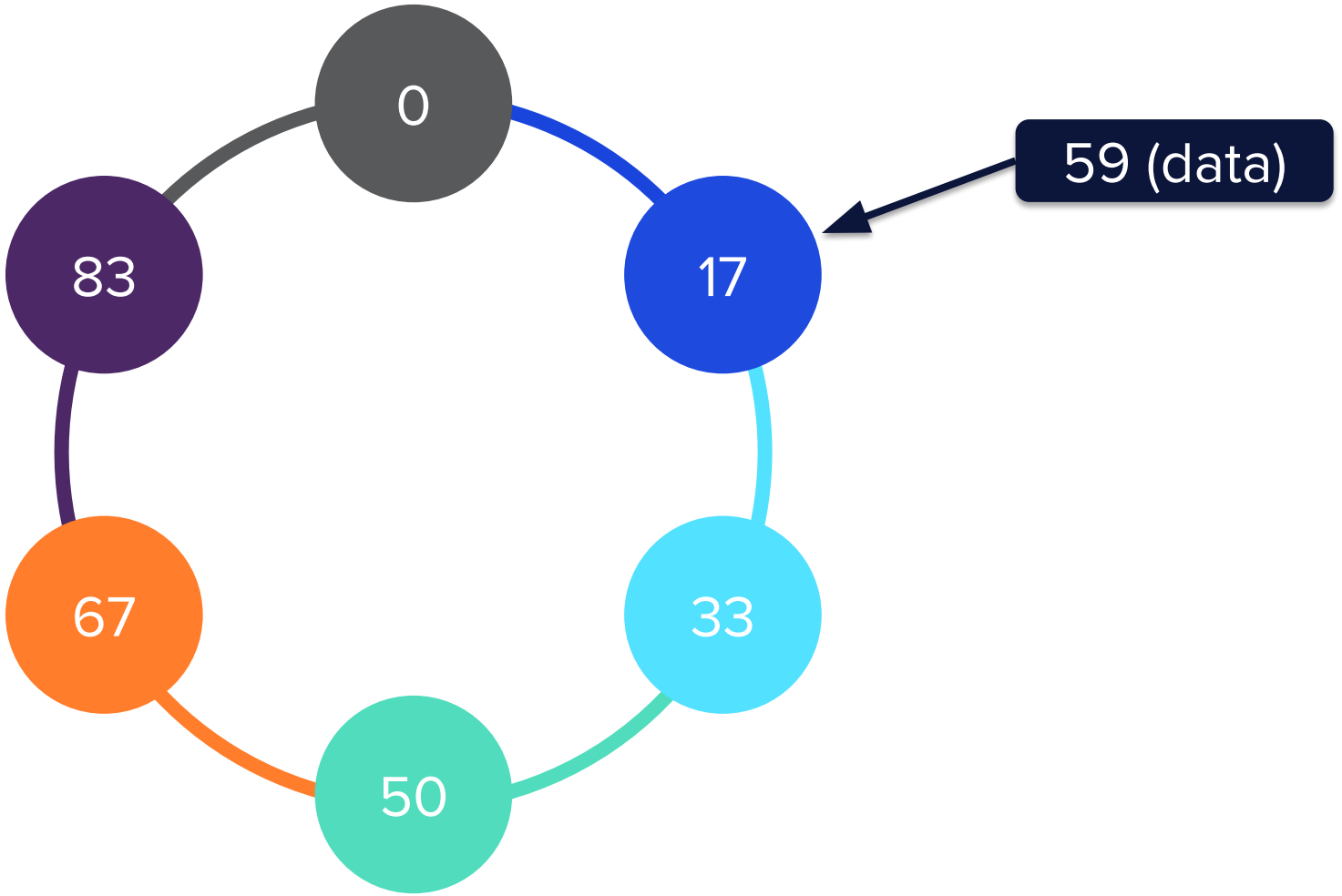


How the Ring Works



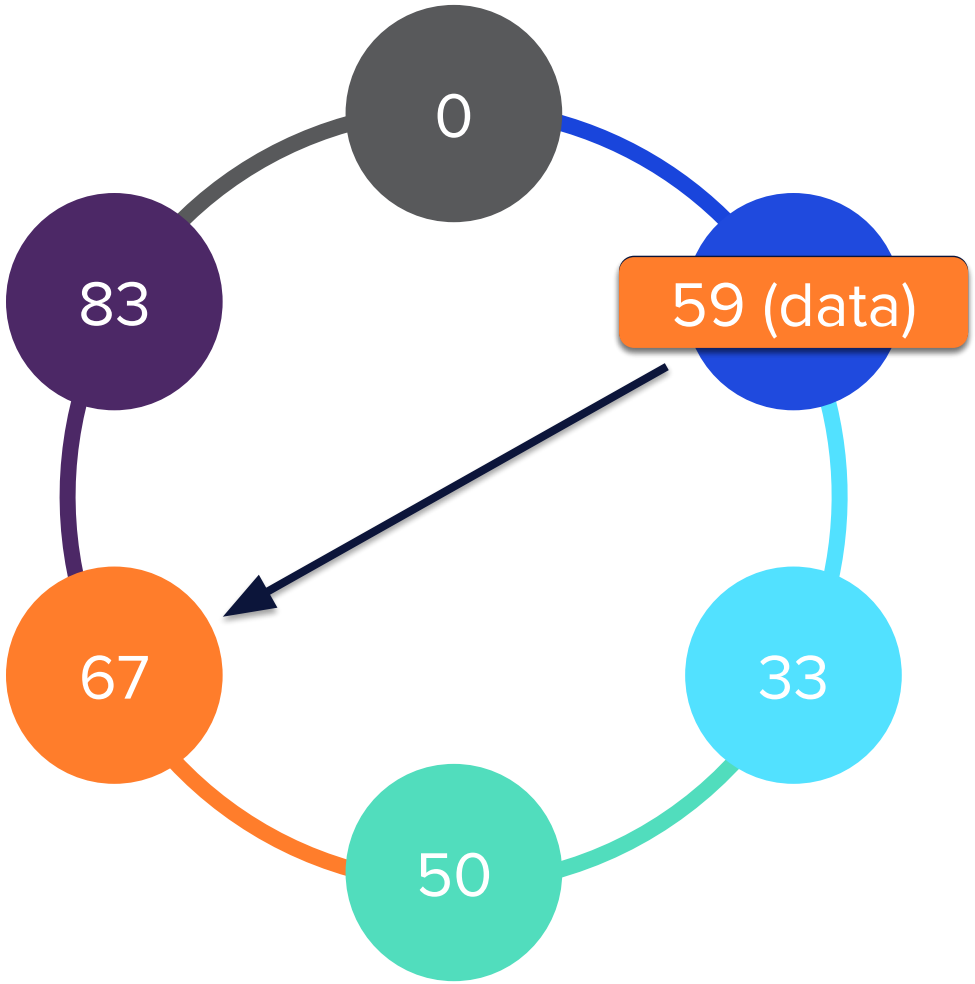
Replication within the Ring

RF = 1



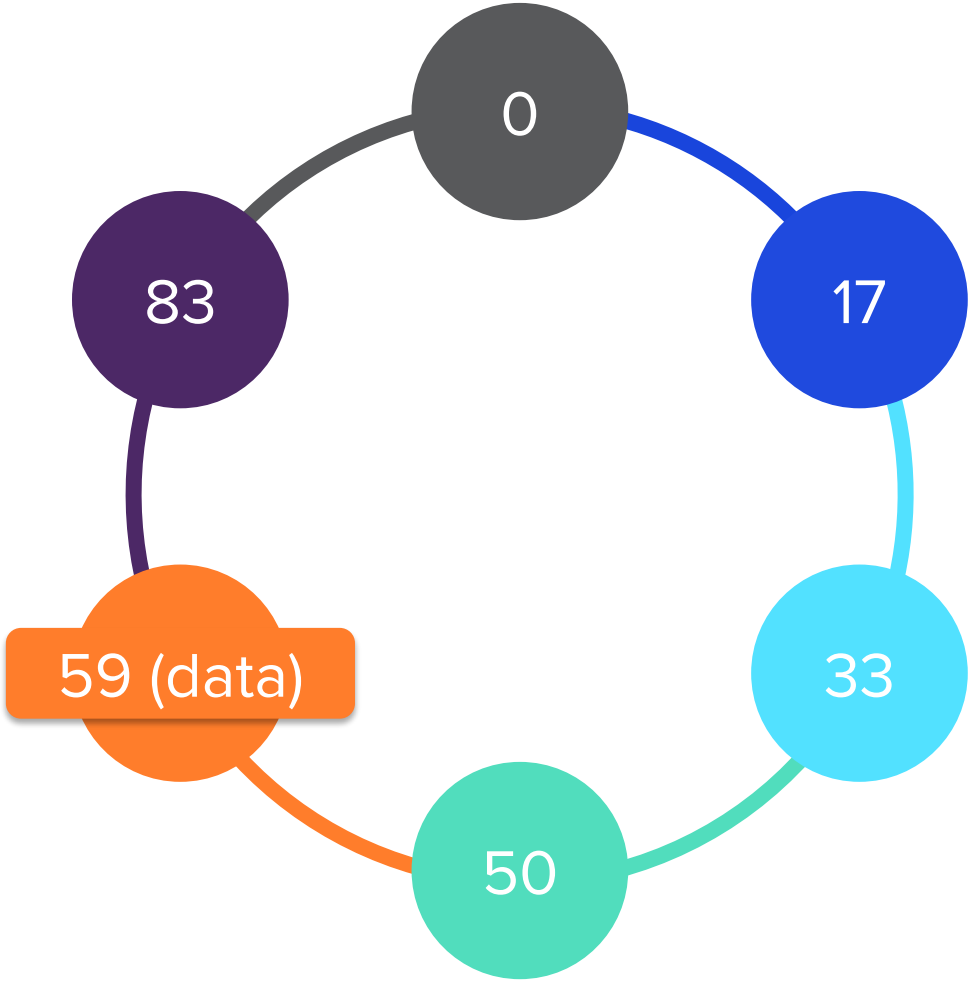
Replication within the Ring

RF = 1



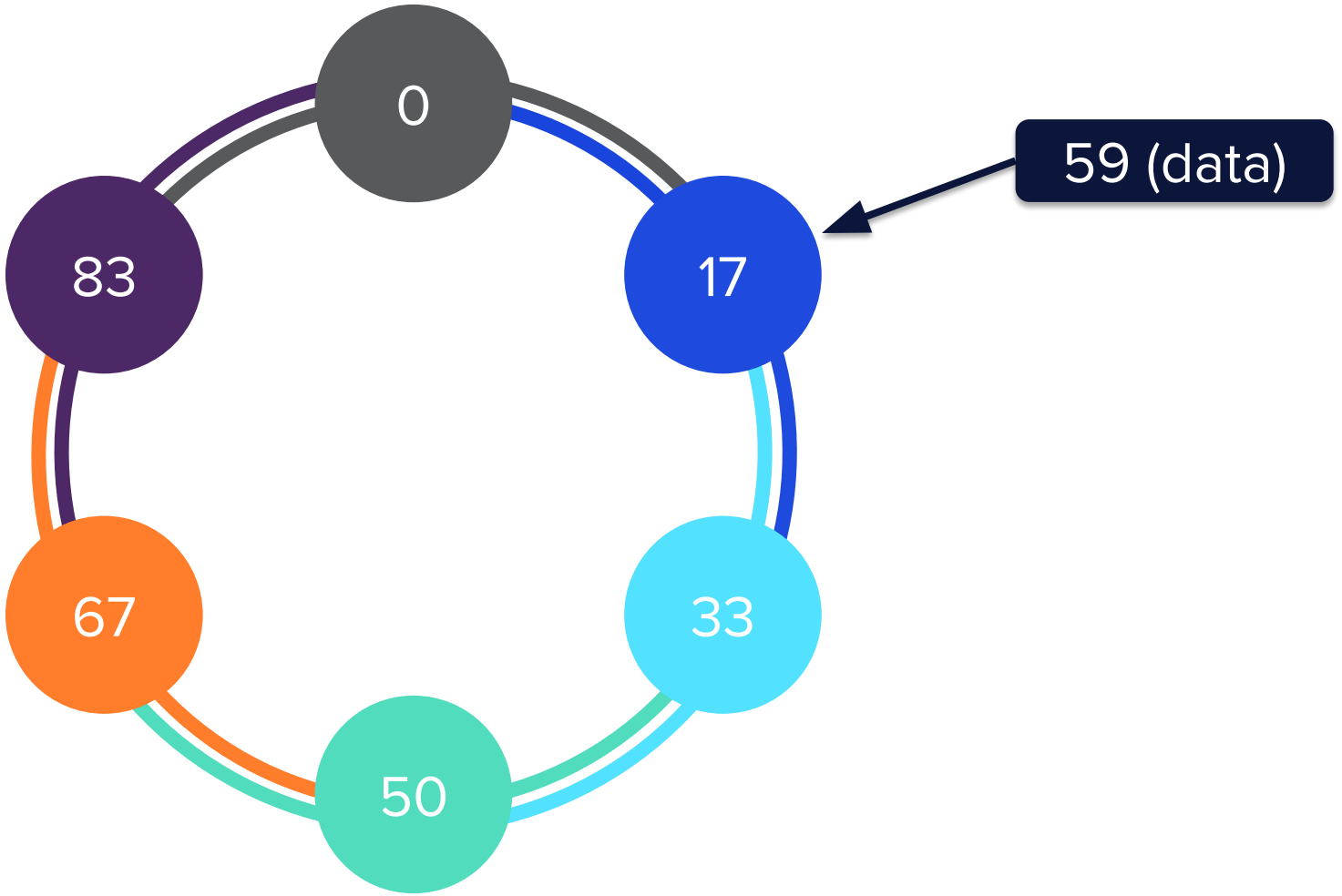
Replication within the Ring

RF = 1



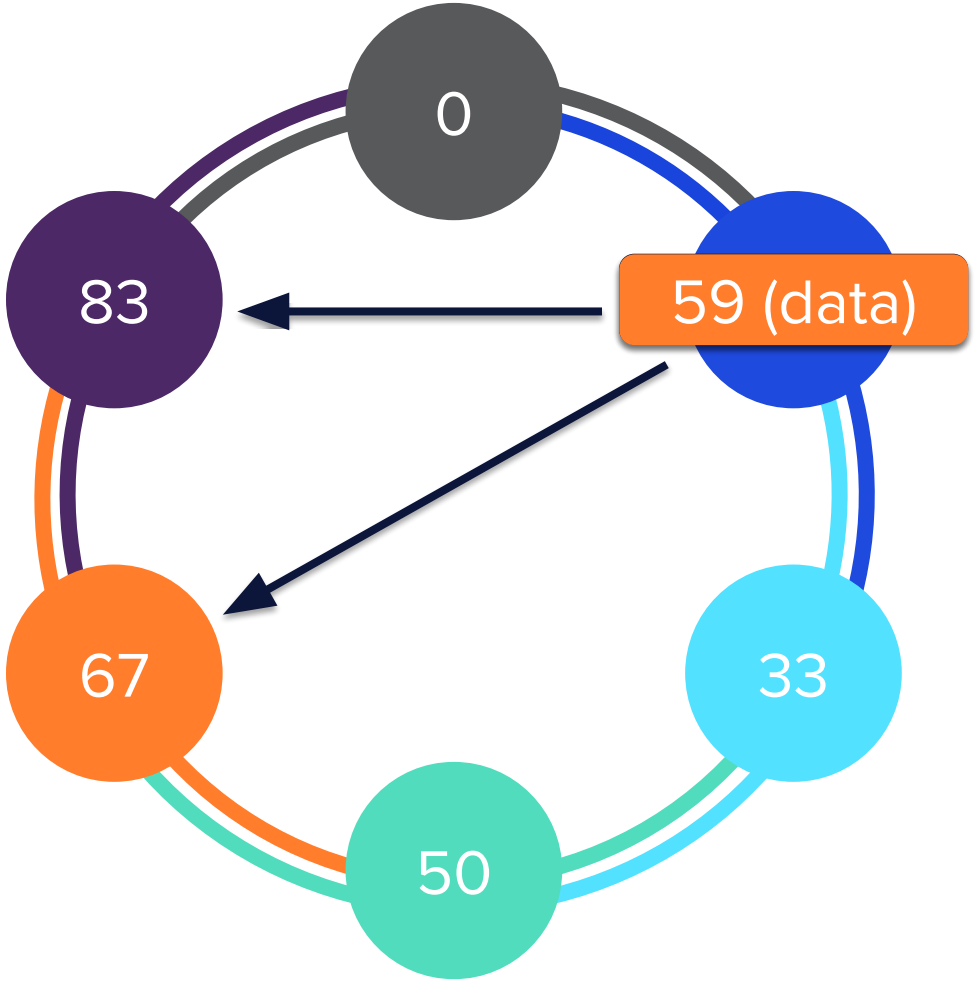
Replication within the Ring

RF = 2



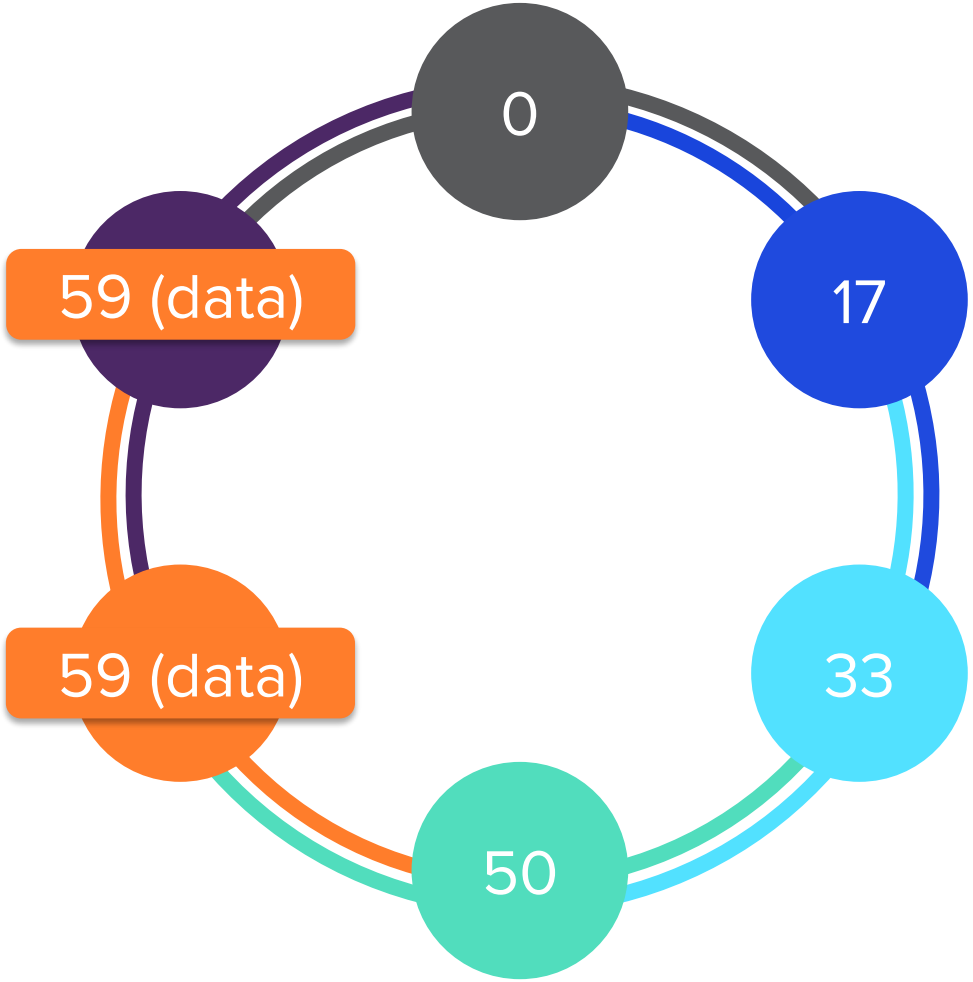
Replication within the Ring

RF = 2



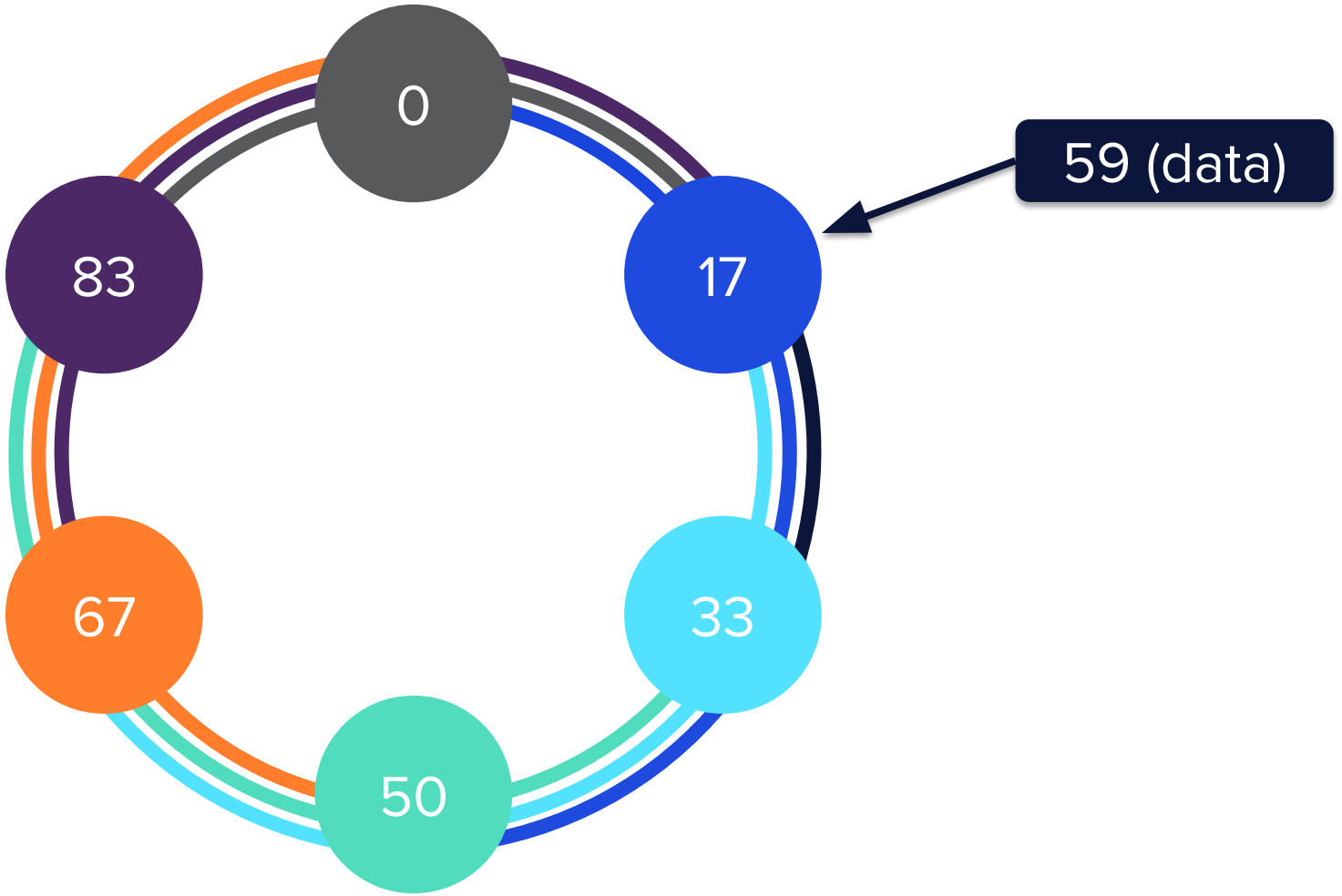
Replication within the Ring

RF = 2



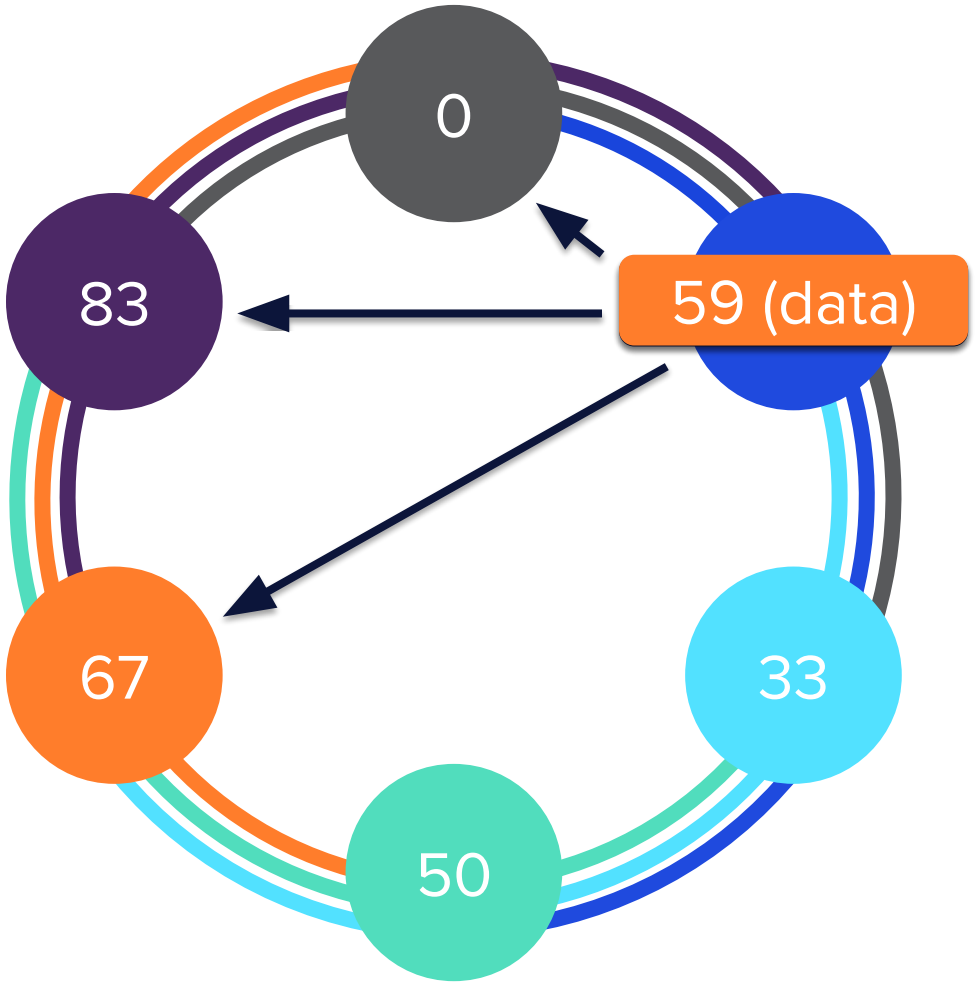
Replication within the Ring

RF = 3



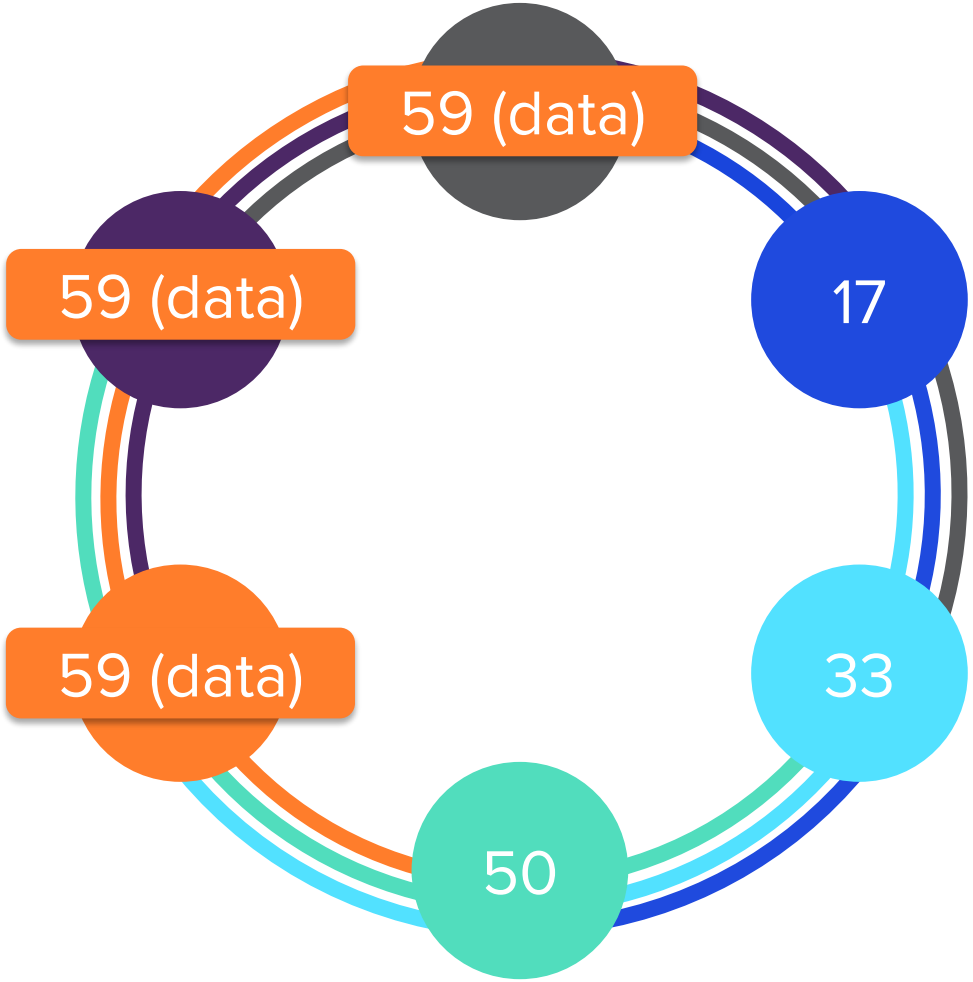
Replication within the Ring

RF = 3



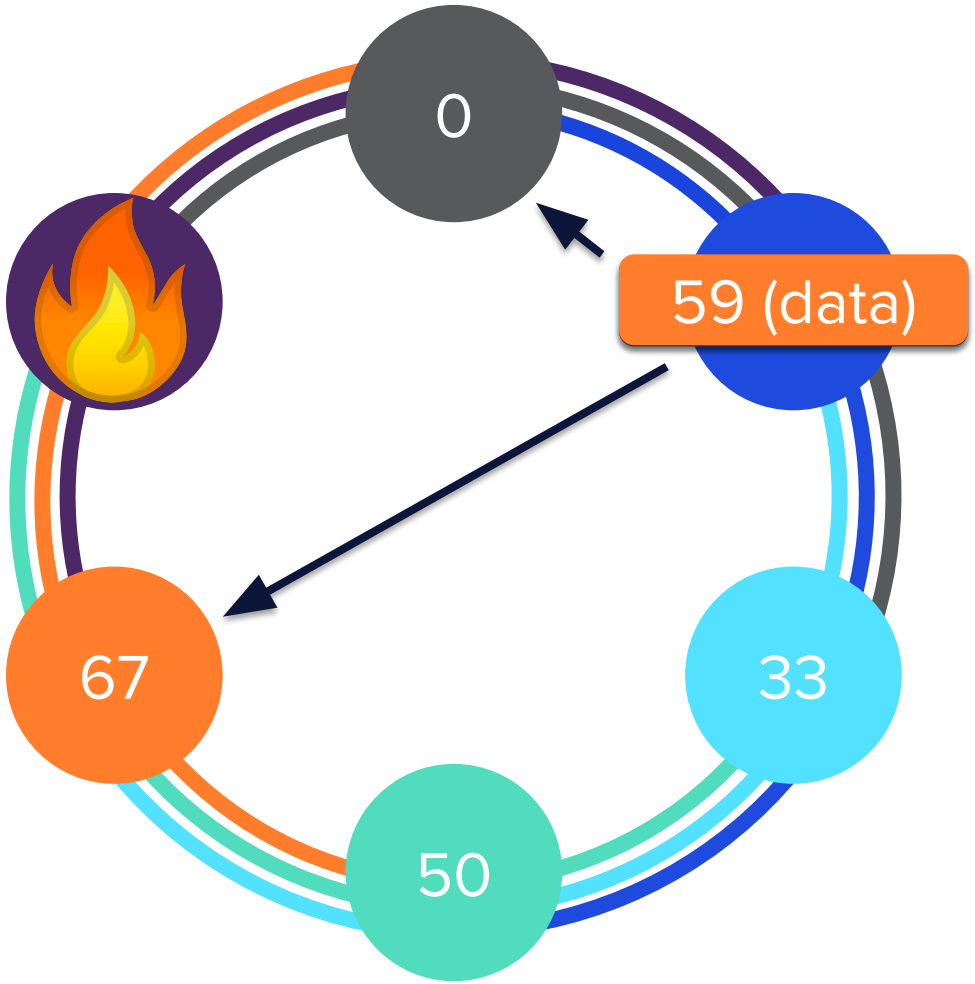
Replication within the Ring

RF = 3



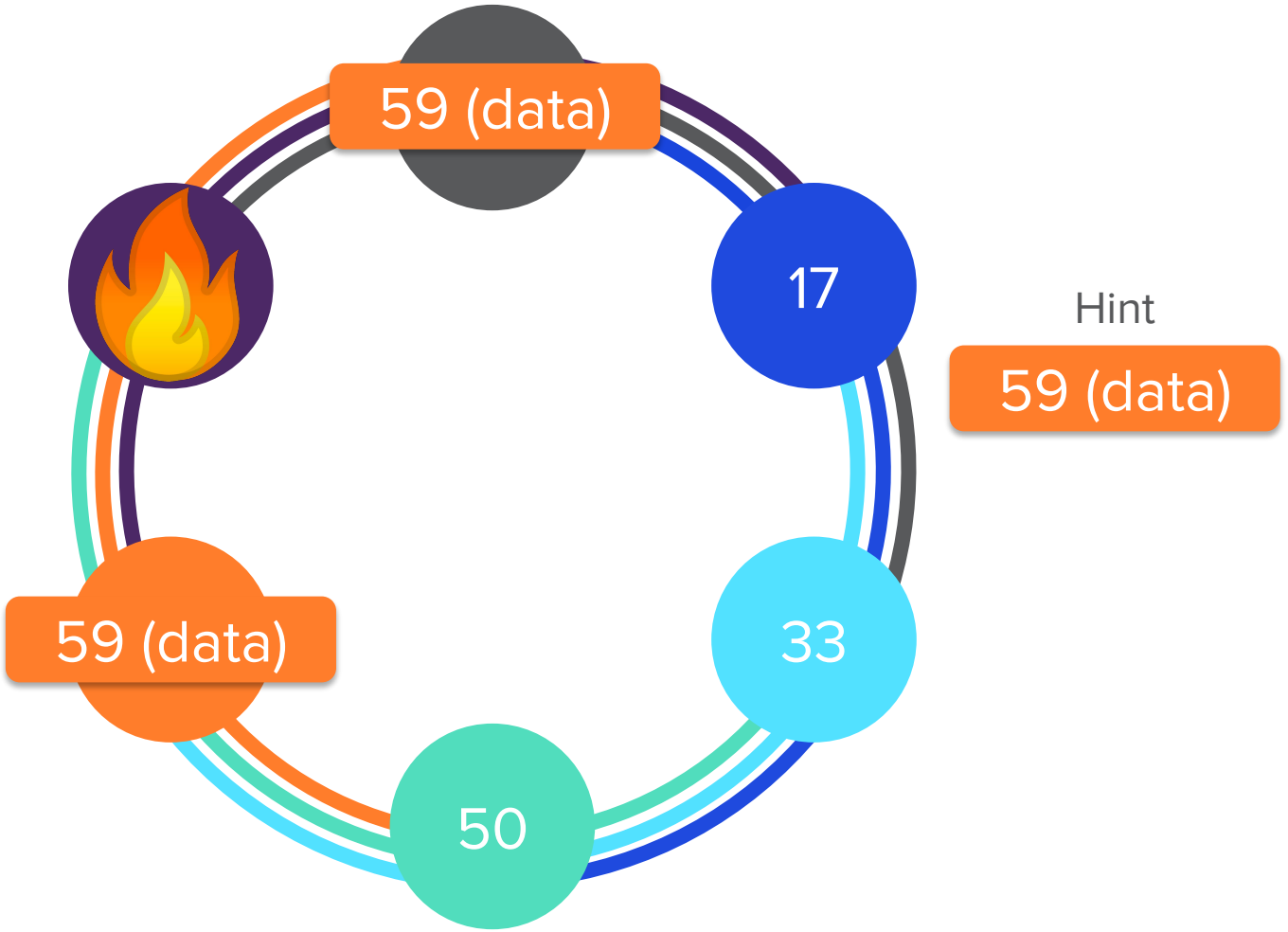
Node Failure

RF = 3



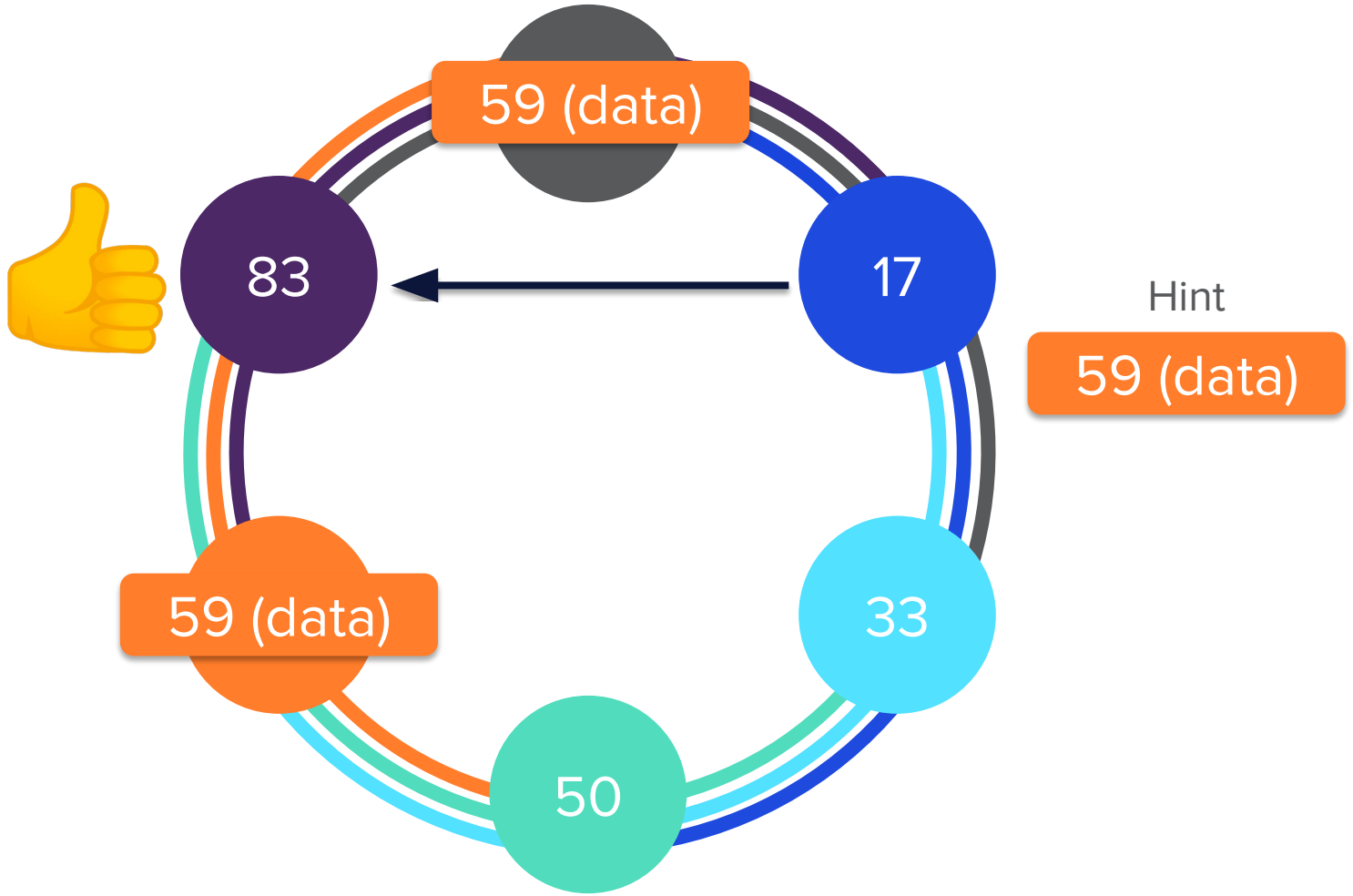
Node Failure

RF = 3



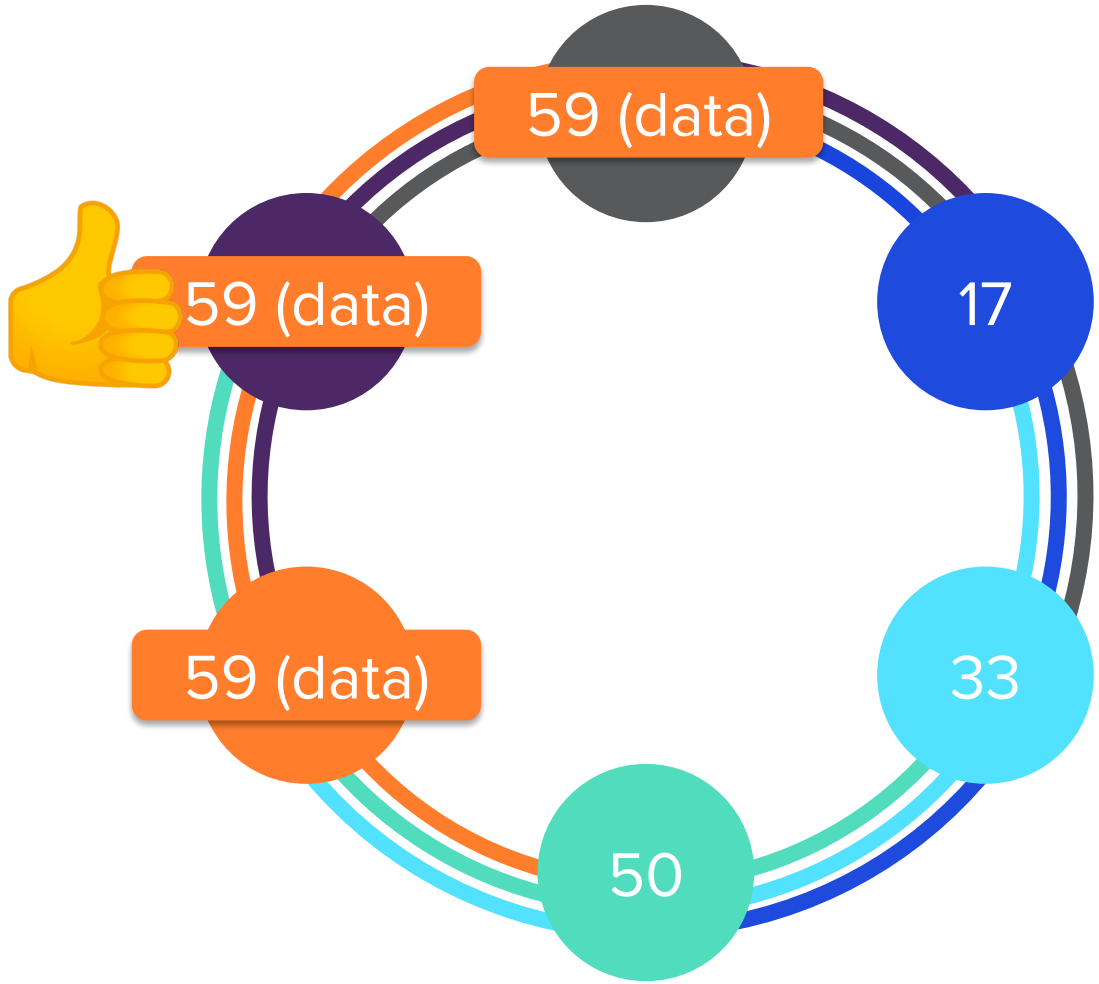
Node Failure

RF = 3

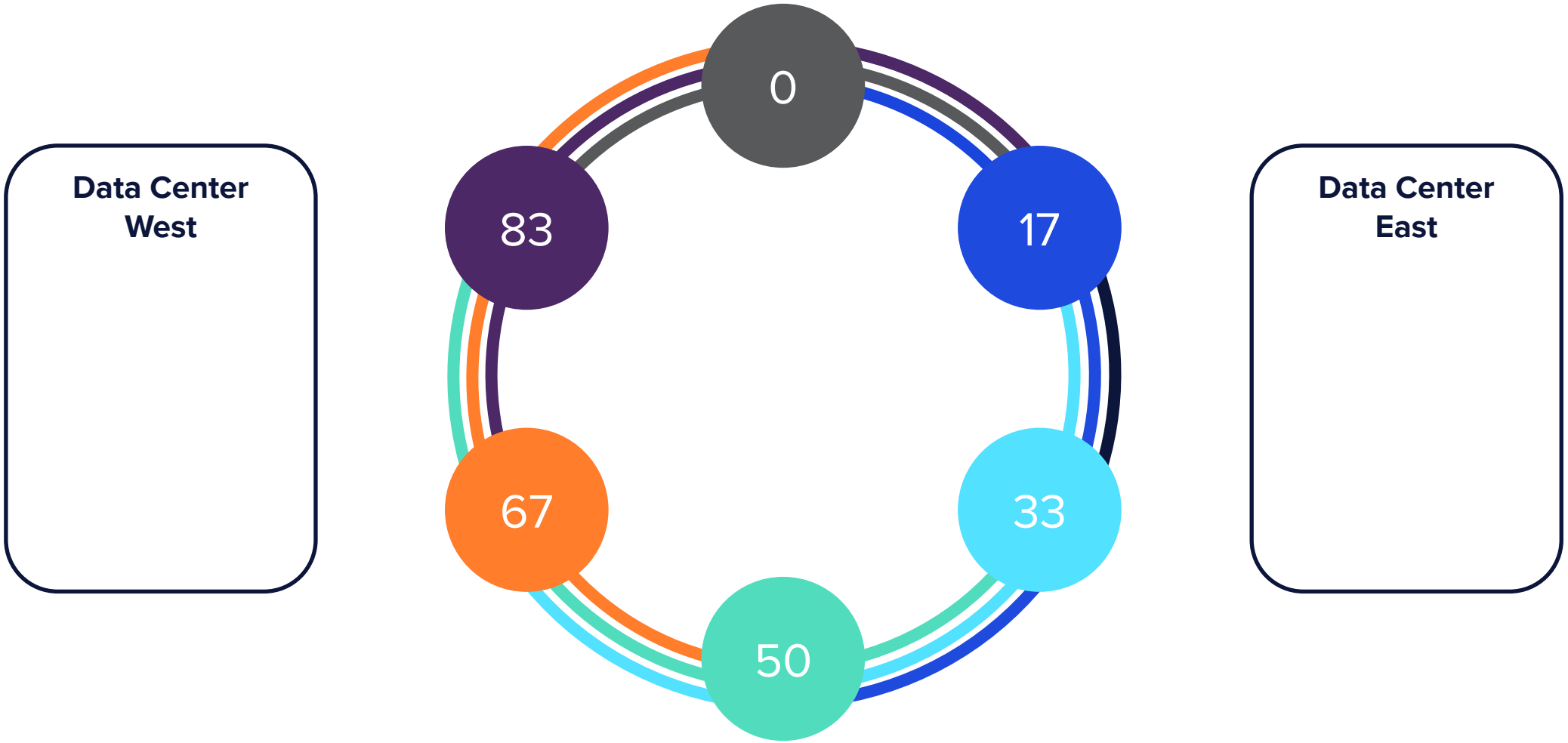


Node Failure – Recovered!

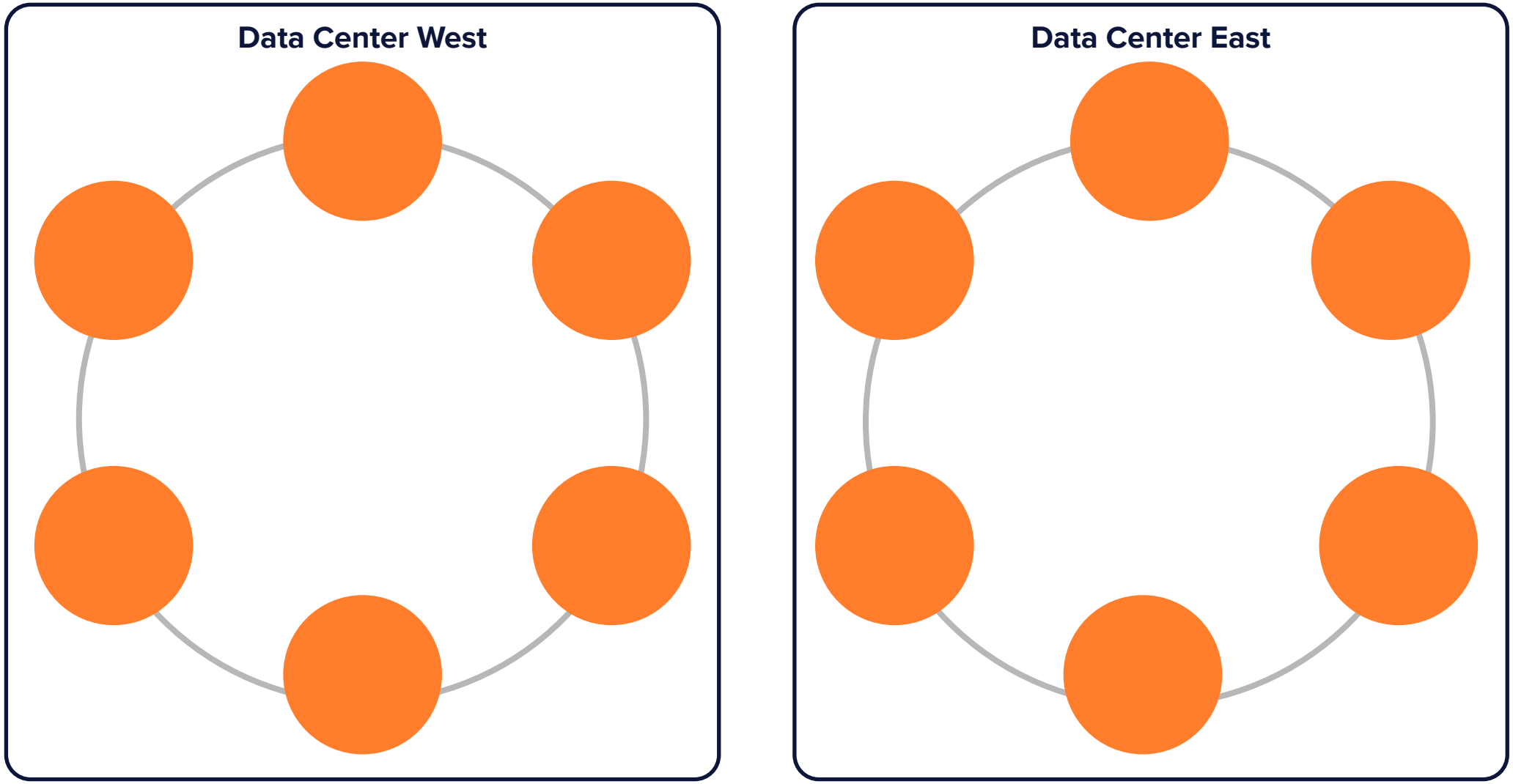
RF = 3



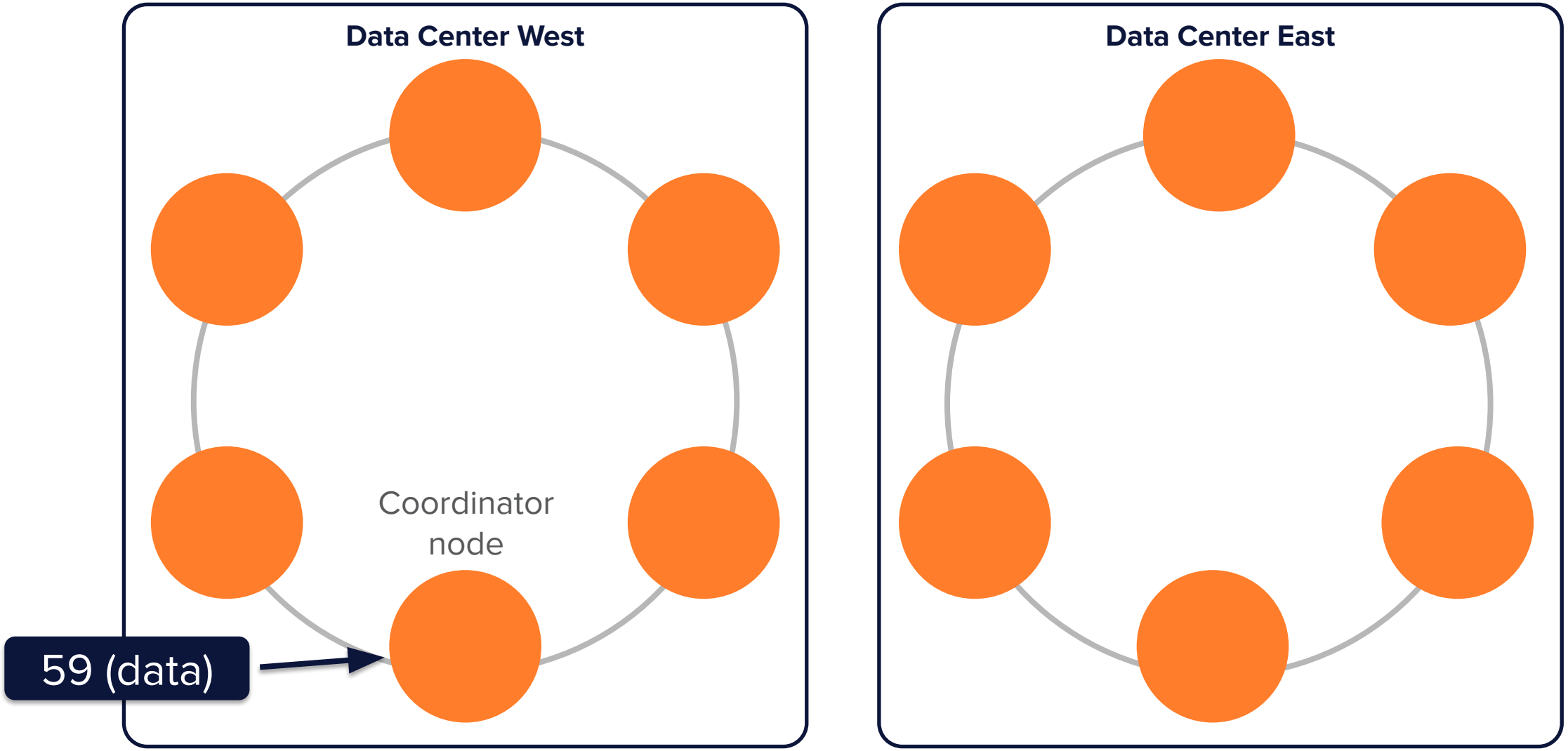
Multi-Data Center Replication



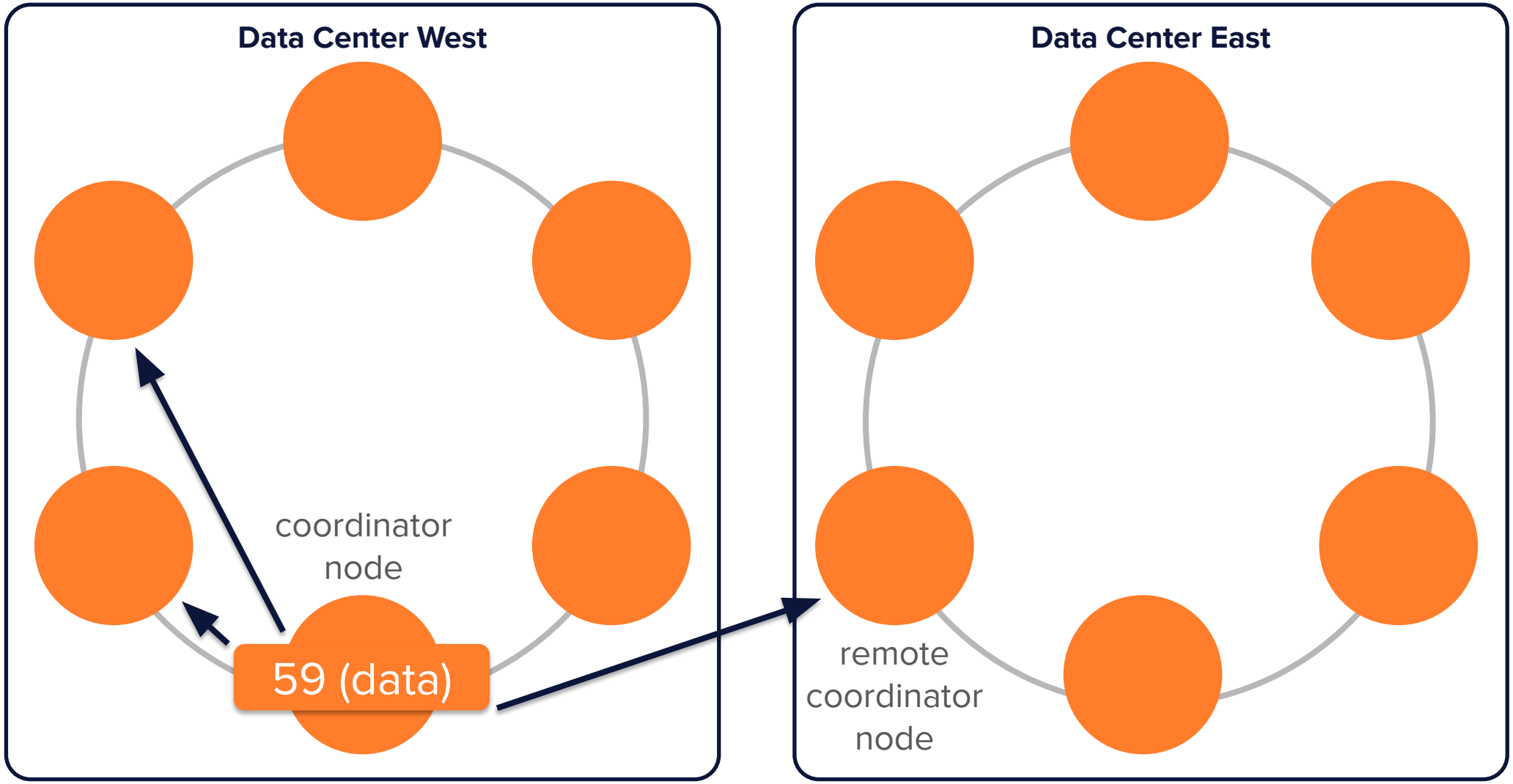
Multi-Data Center Replication (RF=2 in each DC)



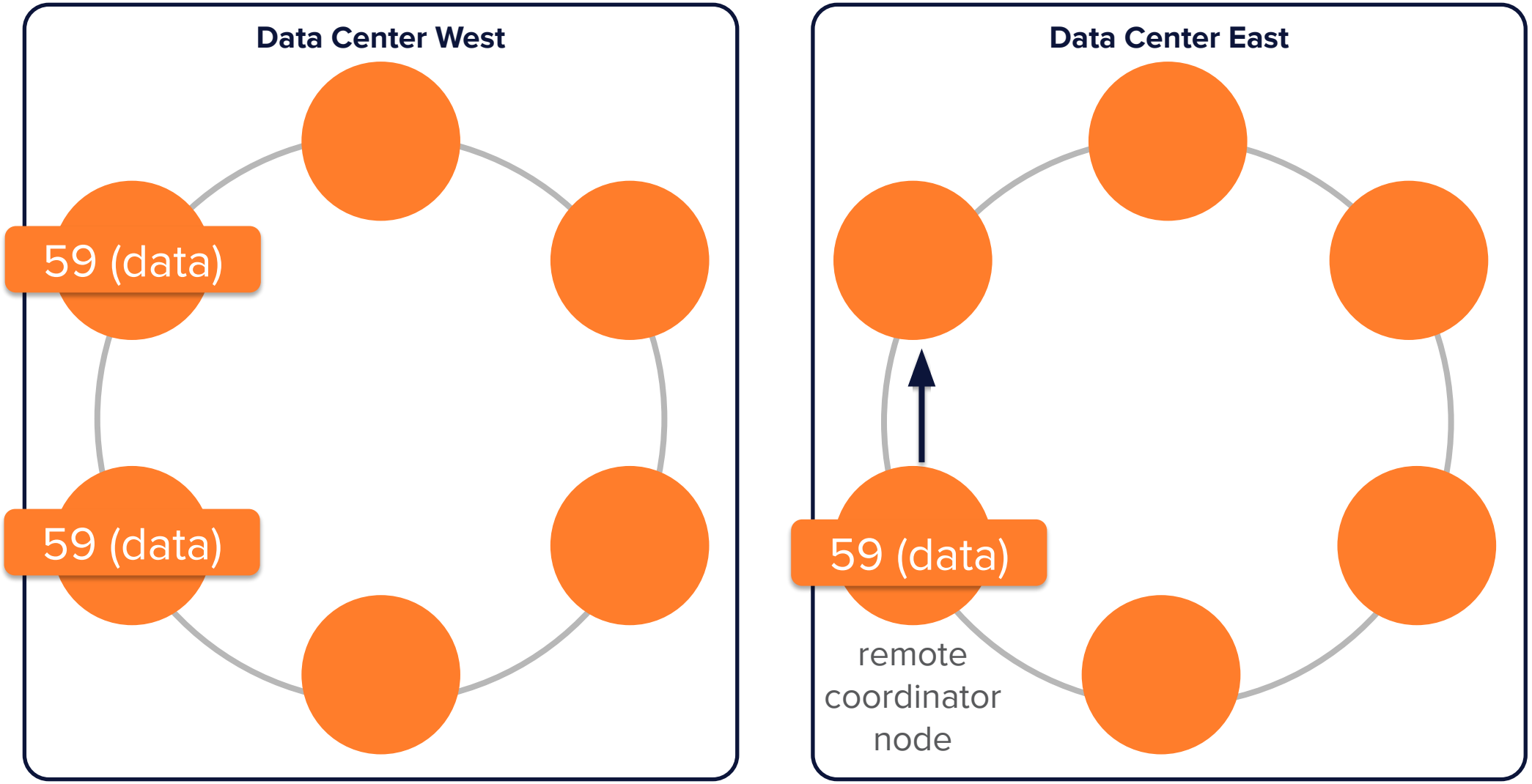
Multi-Data Center Replication (RF=2 in each DC)



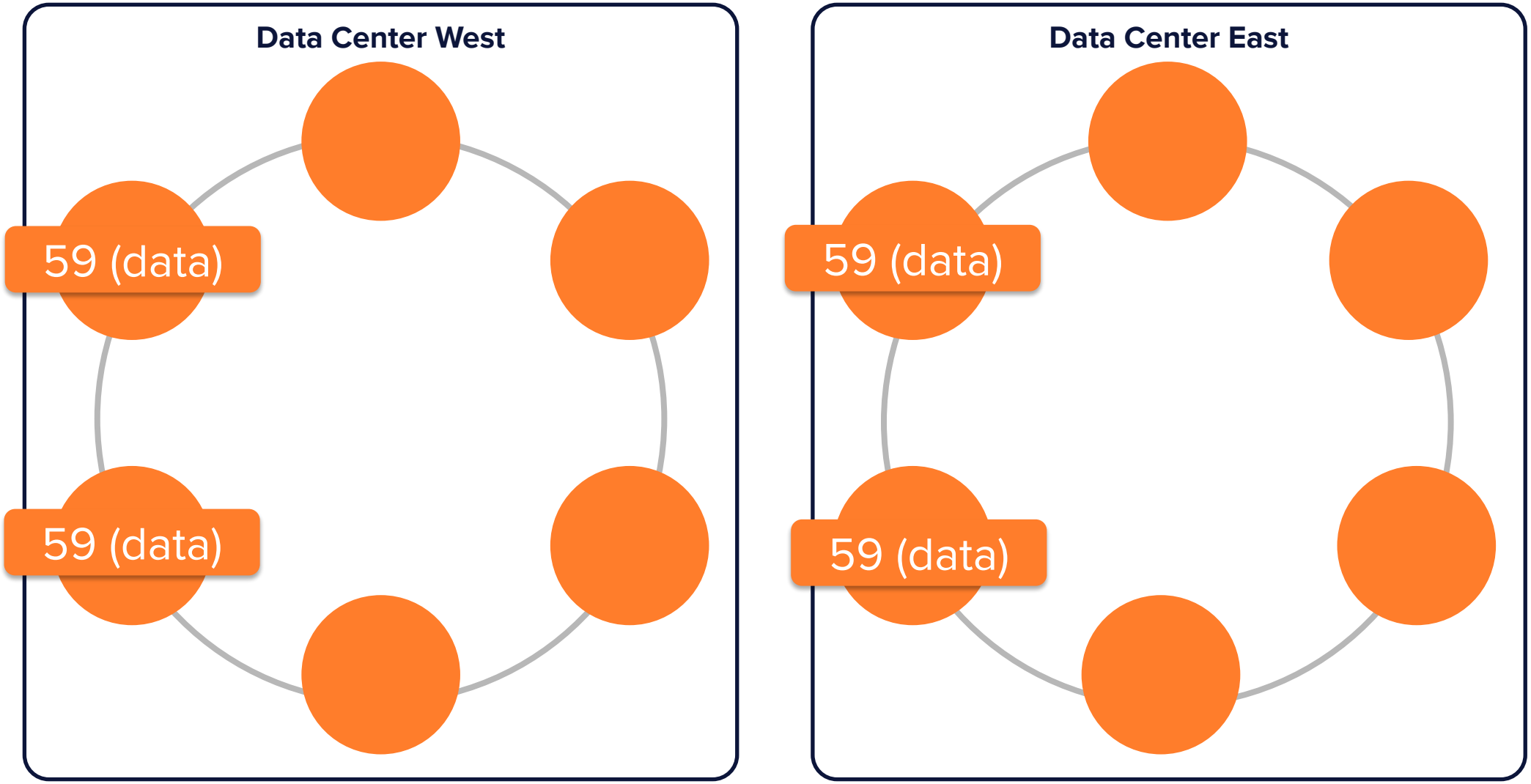
Multi-Data Center Replication (RF=2 in each DC)



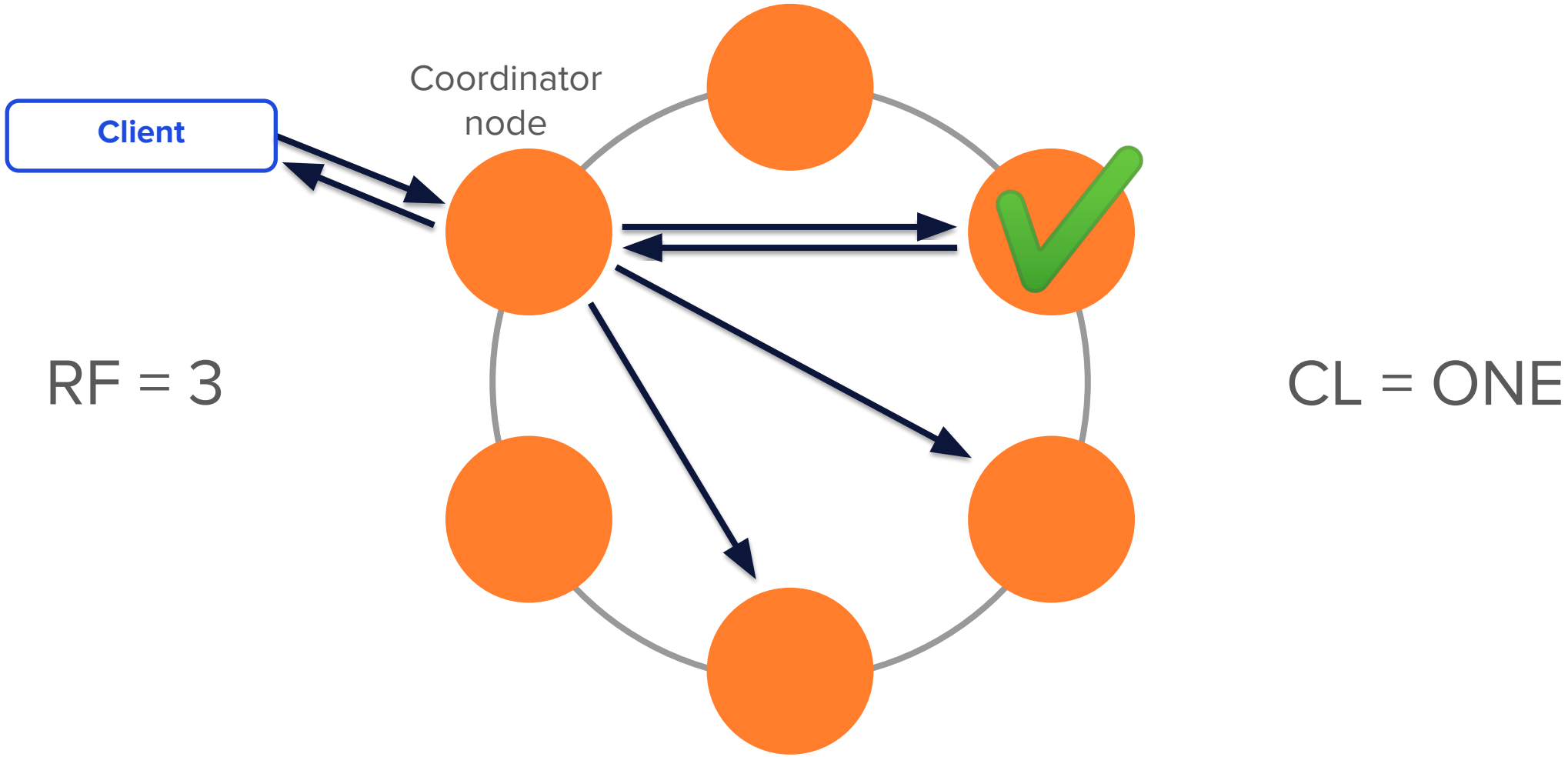
Multi-Data Center Replication (RF=2 in each DC)



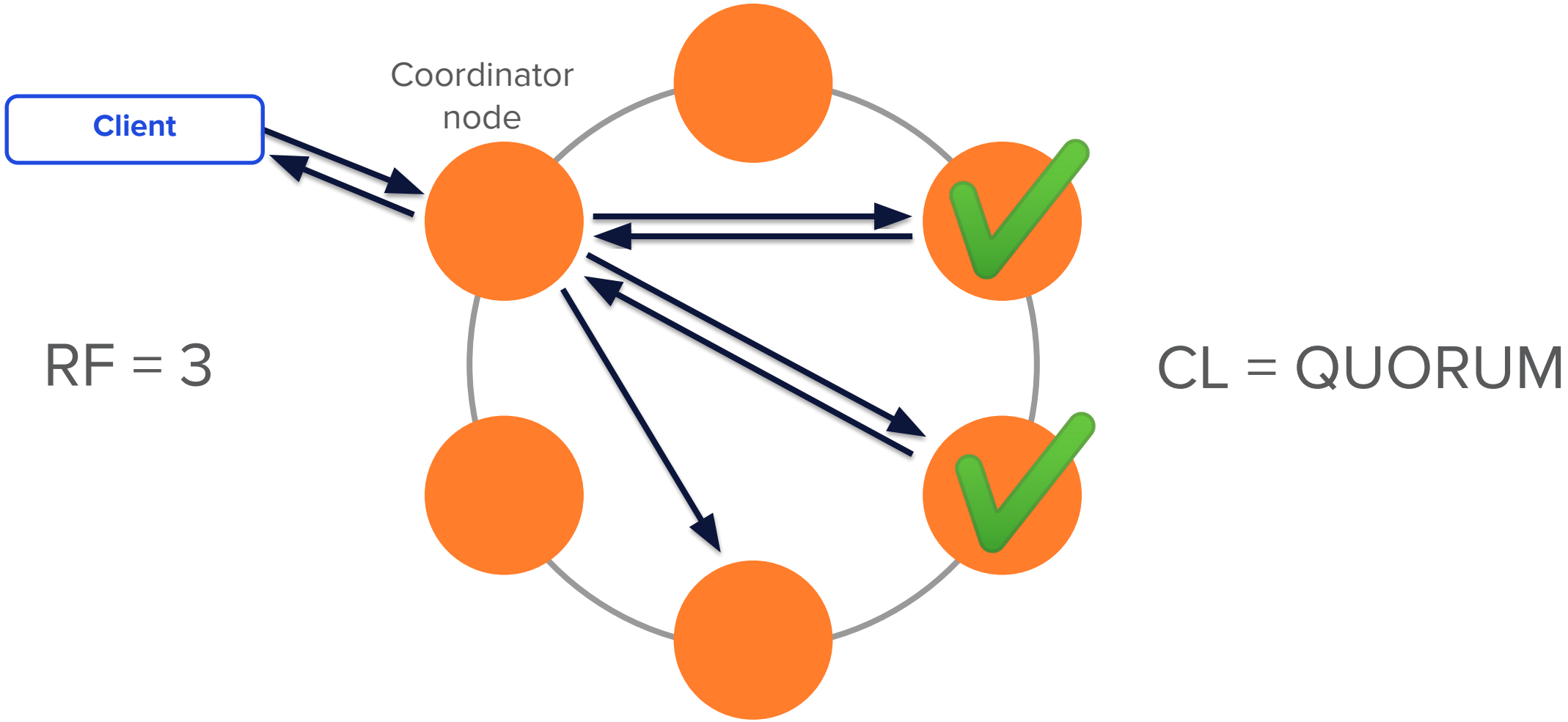
Multi-Data Center Replication (RF=2 in each DC)



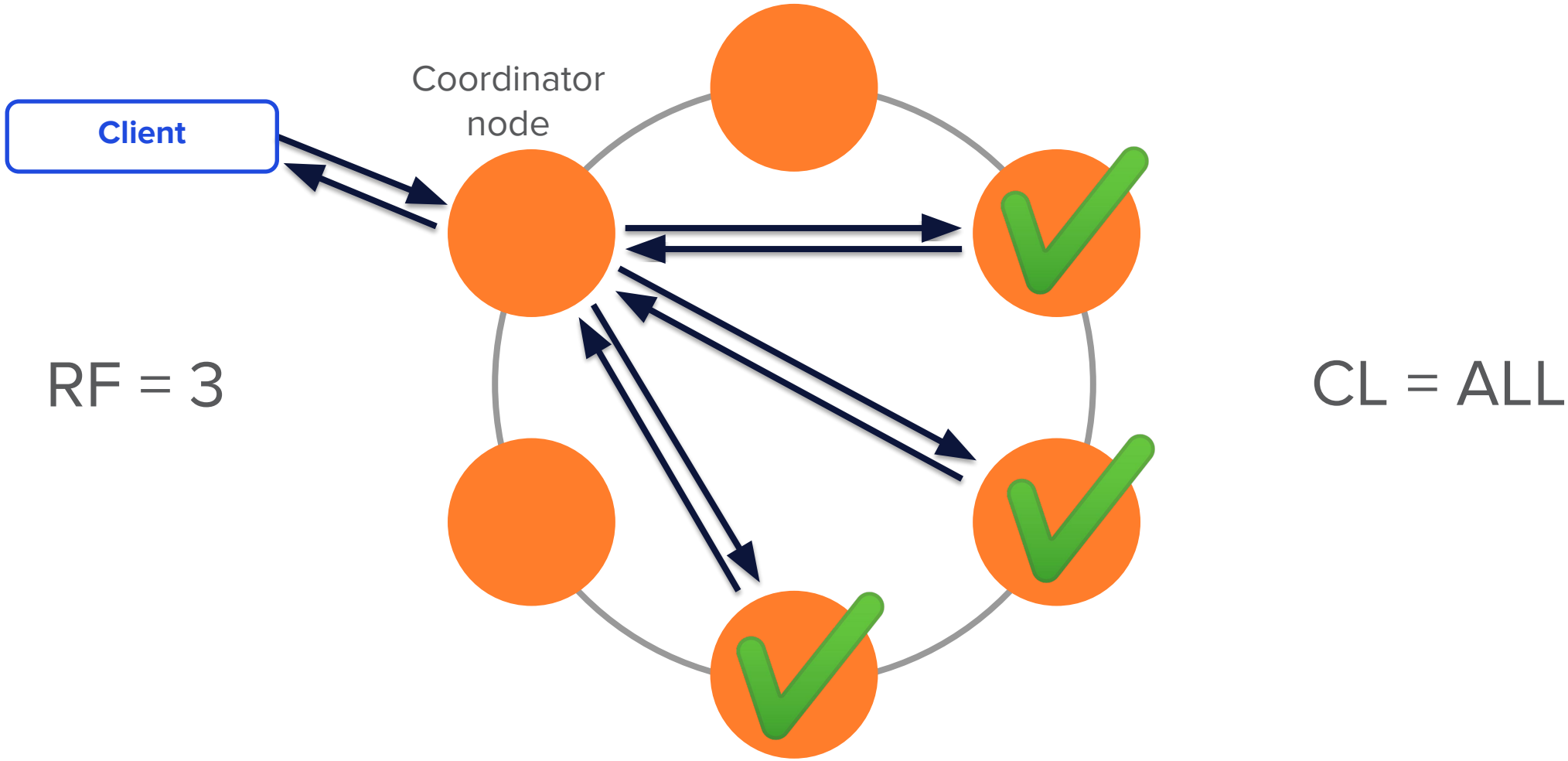
Consistency Levels



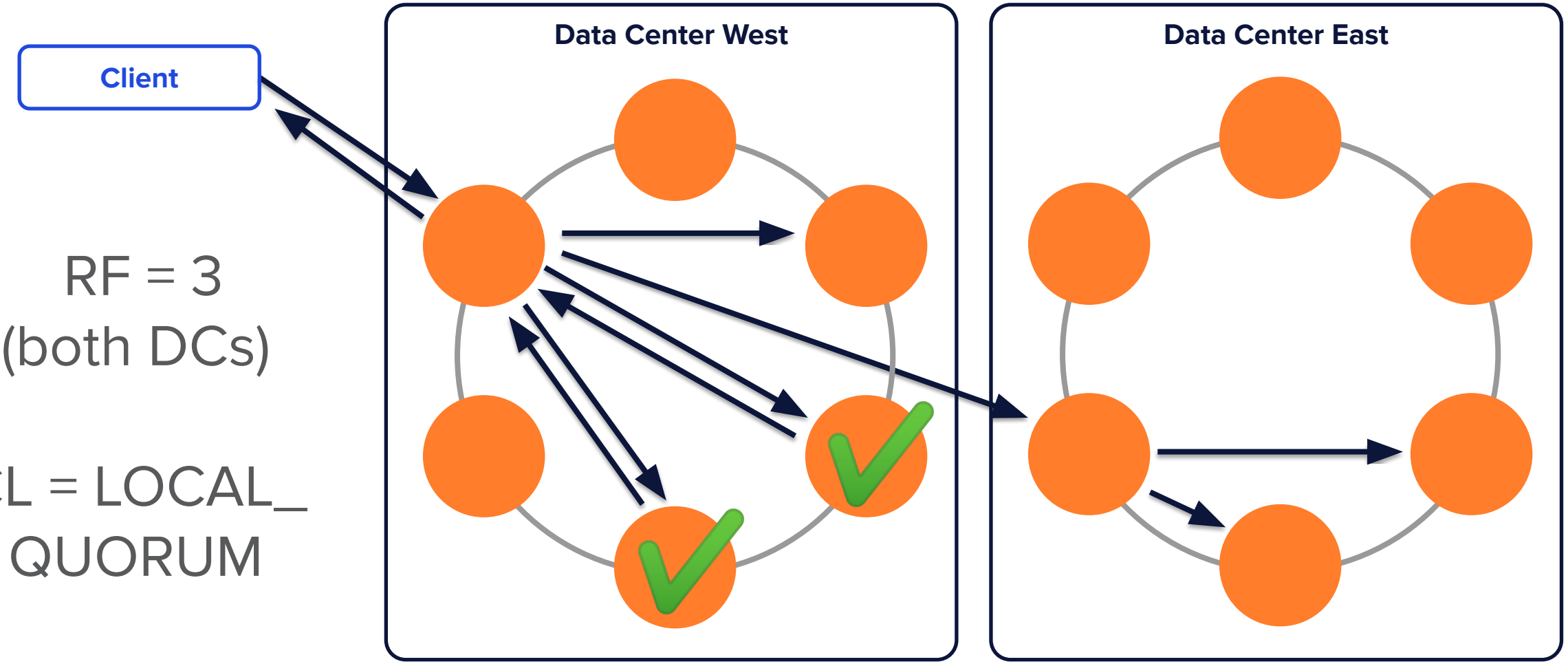
Consistency Levels



Consistency Levels



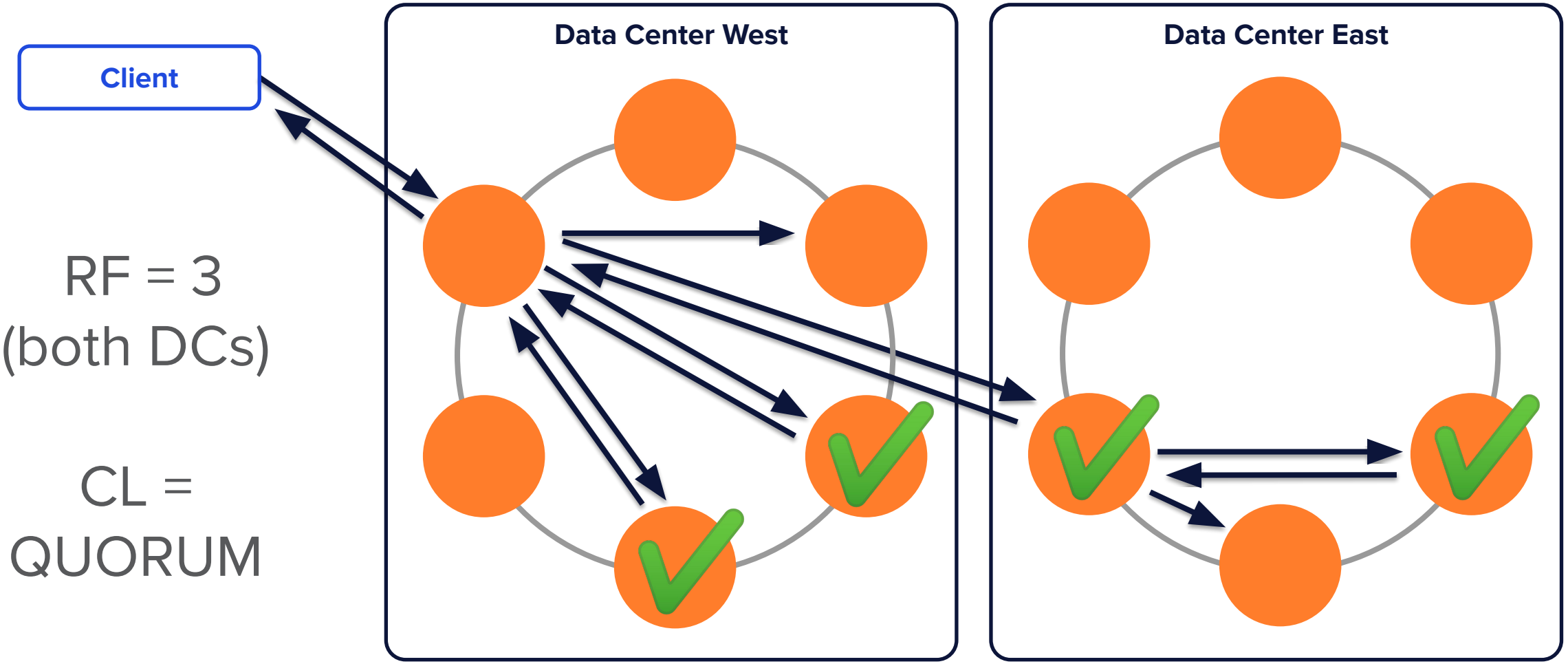
Consistency Across Data Centers



RF = 3
(both DCs)

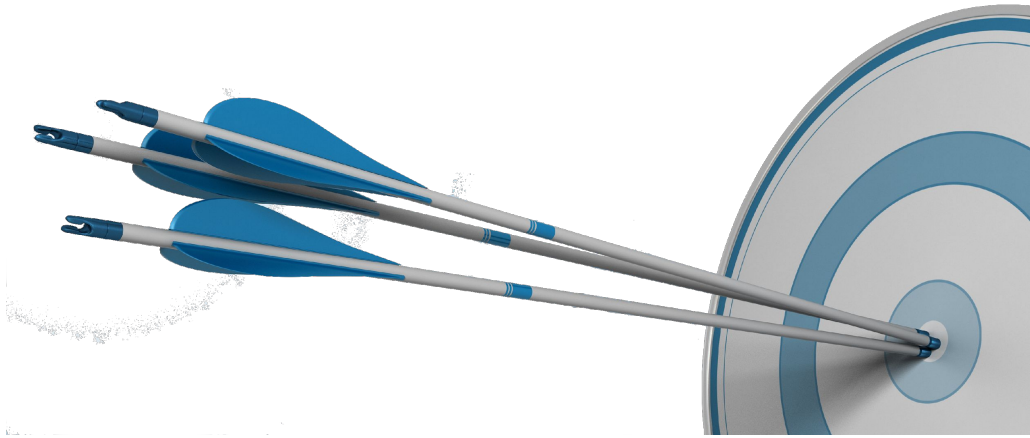
CL = LOCAL_
QUORUM

Consistency Across Data Centers

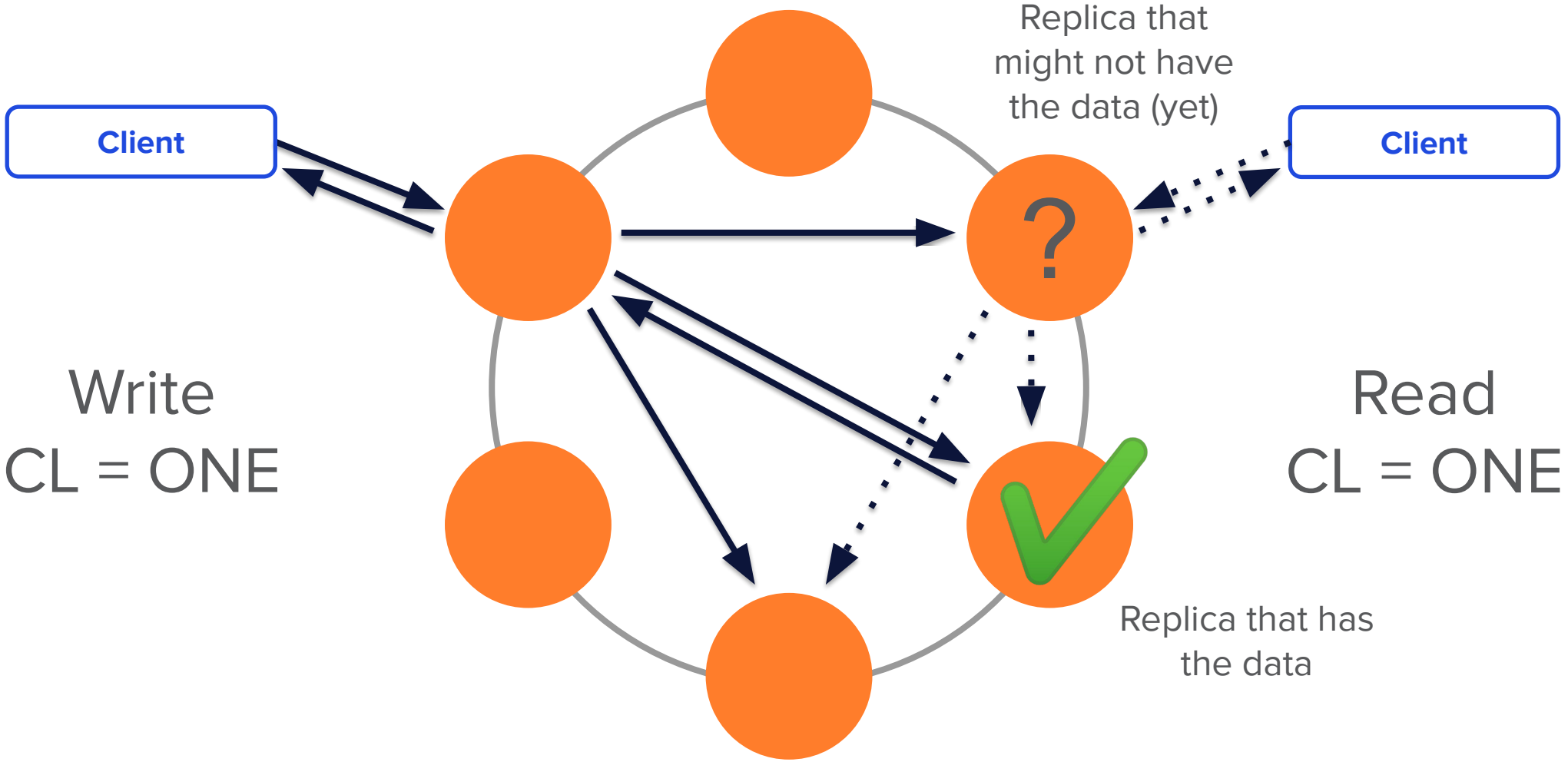


Cassandra Data Availability Concepts

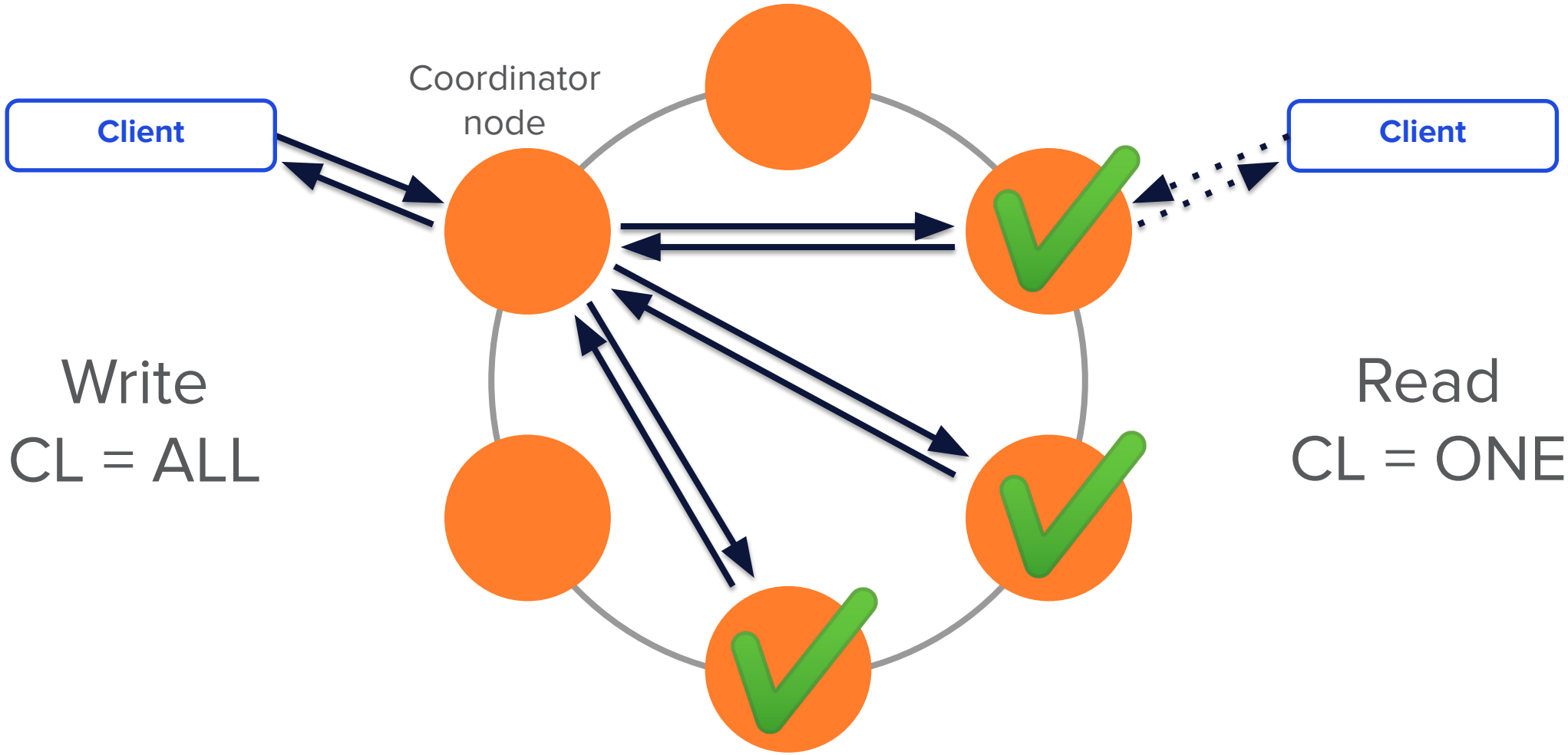
- Replication Factor – the number of copies of your data
  - Increasing replication increases chances of availability
  - Increasing replication increases chances of inconsistency
- Consistency Level – the number of acknowledged copies read/written
- Immediate Consistency: number of writes + number of reads > replication factor
- EXAMPLE:
  - Replication Factor = 3
  - Read/Write Consistency Level = QUORUM
  - 2 + 2 = 4 > 3 👍



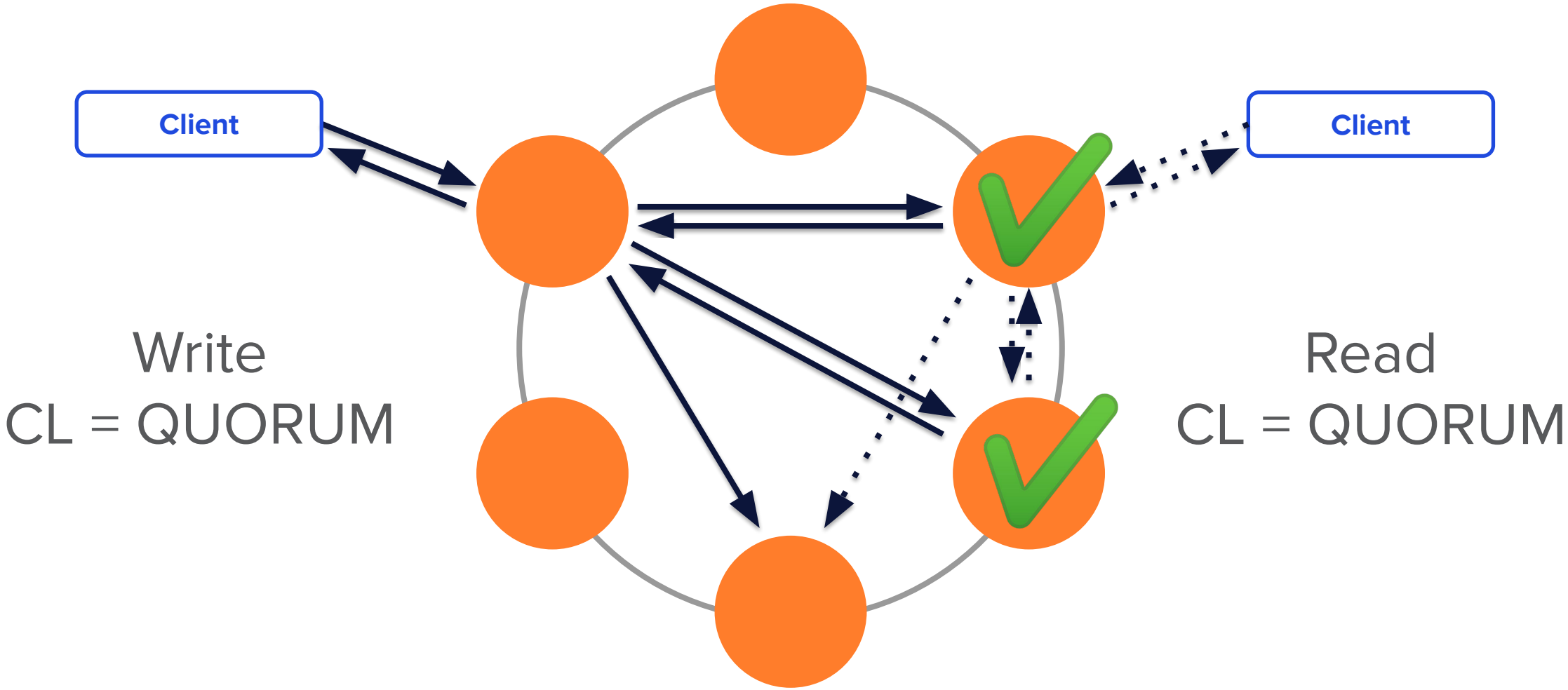
Weak(er) Consistency



Immediate Consistency – One Way



Immediate Consistency – A Better Way



Consistency Settings

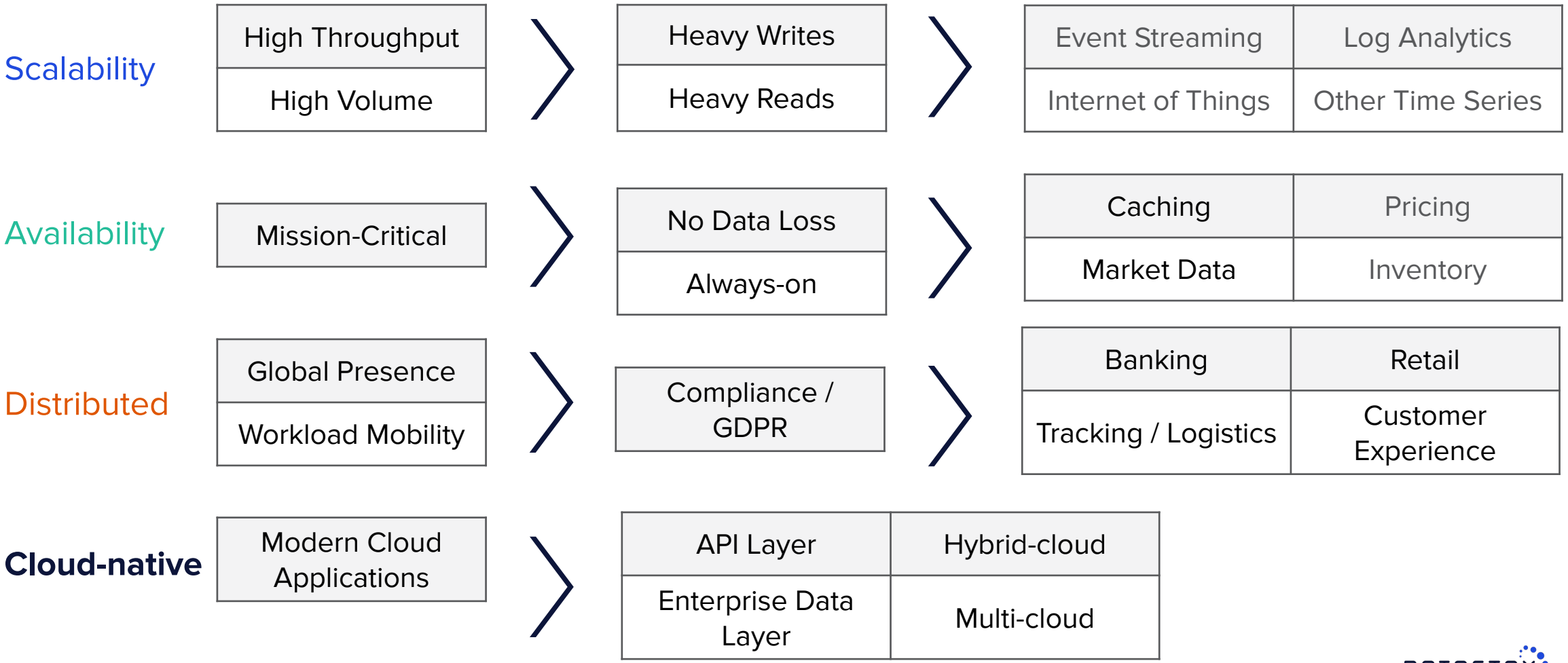
- Weakest to strongest

Setting	Description
ANY	Storing a hint at minimum is satisfactory
ONE, TWO, THREE	Checks closest node(s) to coordinator
QUORUM	Majority vote, (sum_of_replication_factors / 2) + 1
LOCAL_ONE	Closest node to coordinator in same data center
LOCAL_QUORUM	Closest quorum of nodes in same data center
EACH_QUORUM	Quorum of nodes in each data center, applies to writes only
ALL	Every node must participate

Consistency Tradeoffs

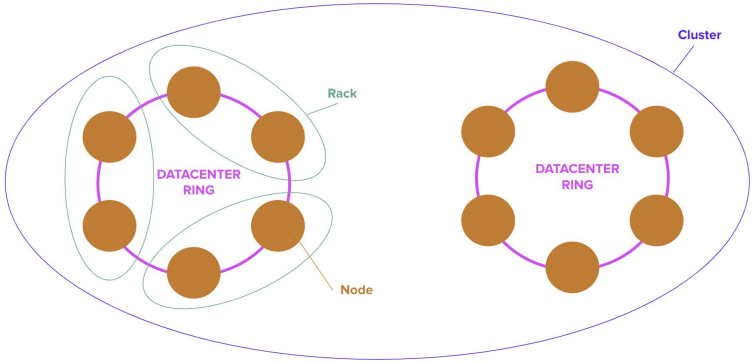
- The higher the consistency, the less chance you may get stale data
  - Pay for this with POSSIBLE latency
  - Depends on your situational needs
- IoT and Sensor systems with write-heavy workloads may benefit from CL=ONE for writes if data isn't required immediately

Understanding Use Cases

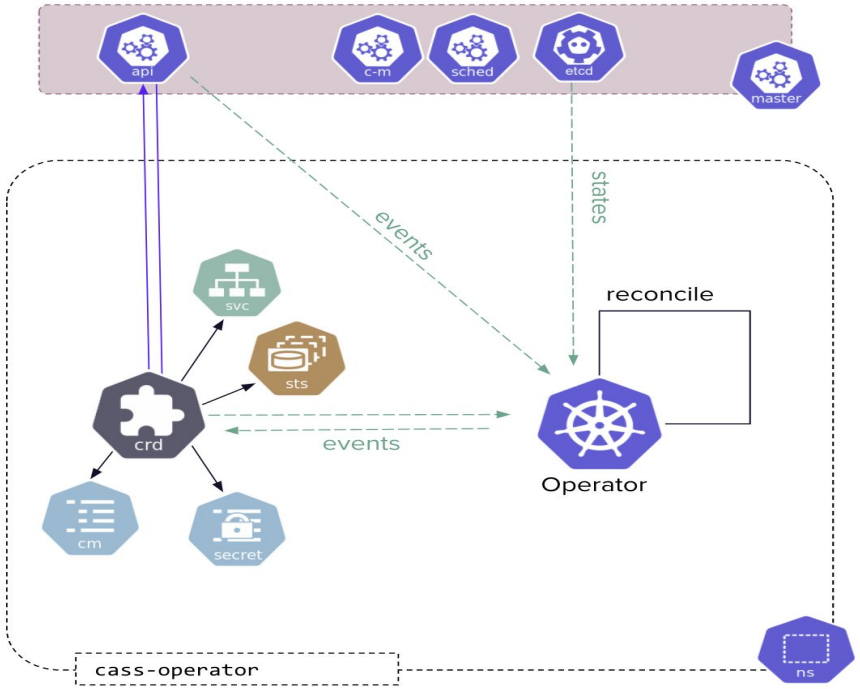


DATASTAX

YOUR MISSION
SHOULD YOU
CHOOSE TO
ACCEPT IT



cassandra



kubernetes

Online Workshops

DATASTAX

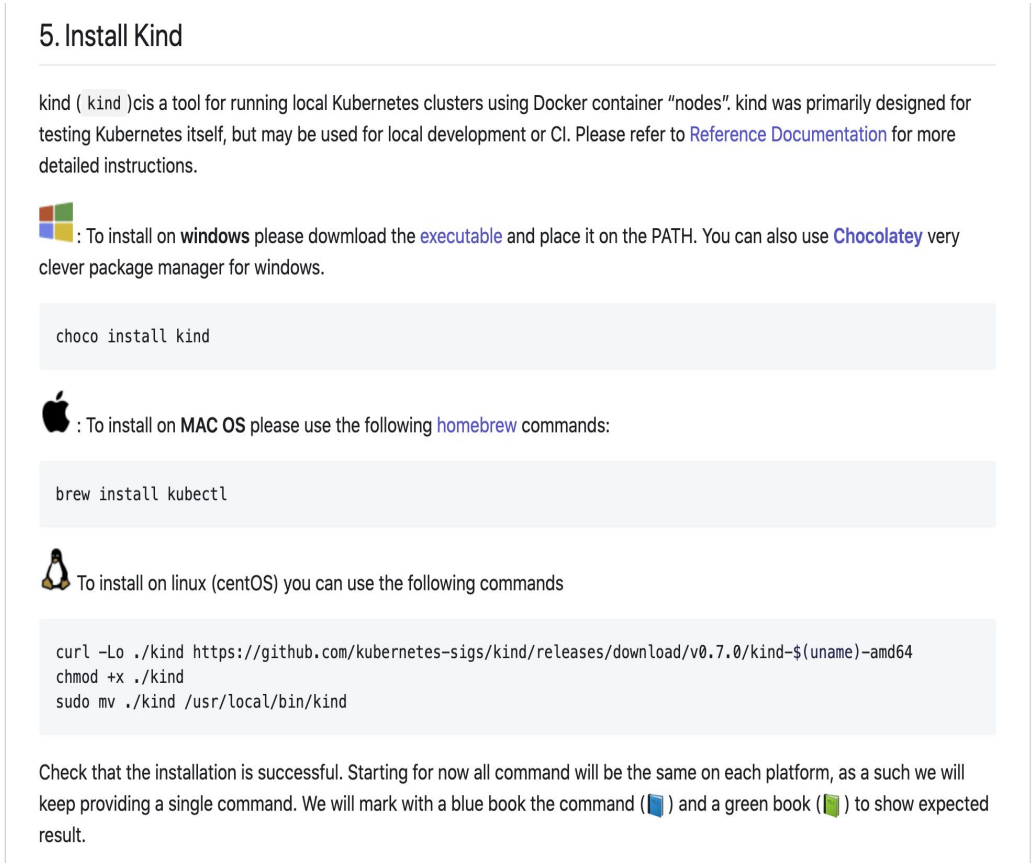
Section #0 :
Setup Your Cluster

https://github.com/DataStax-Academy/kubecon-cassandra-workshop

- Your Laptop

  OR

- Our Cloud Instance



5. Install Kind

kind (`kind`)cis a tool for running local Kubernetes clusters using Docker container "nodes". kind was primarily designed for testing Kubernetes itself, but may be used for local development or CI. Please refer to Reference Documentation for more detailed instructions.

: To install on **windows** please dowmload the executable and place it on the PATH. You can also use **Chocolatey** very clever package manager for windows.

```
choco install kind
```

: To install on **MAC OS** please use the following homebrew commands:

```
brew install kubectl
```

To install on linux (centOS) you can use the following commands

```
curl -Lo ./kind https://github.com/kubernetes-sigs/kind/releases/download/v0.7.0/kind-$(uname)-amd64
chmod +x ./kind
sudo mv ./kind /usr/local/bin/kind
```

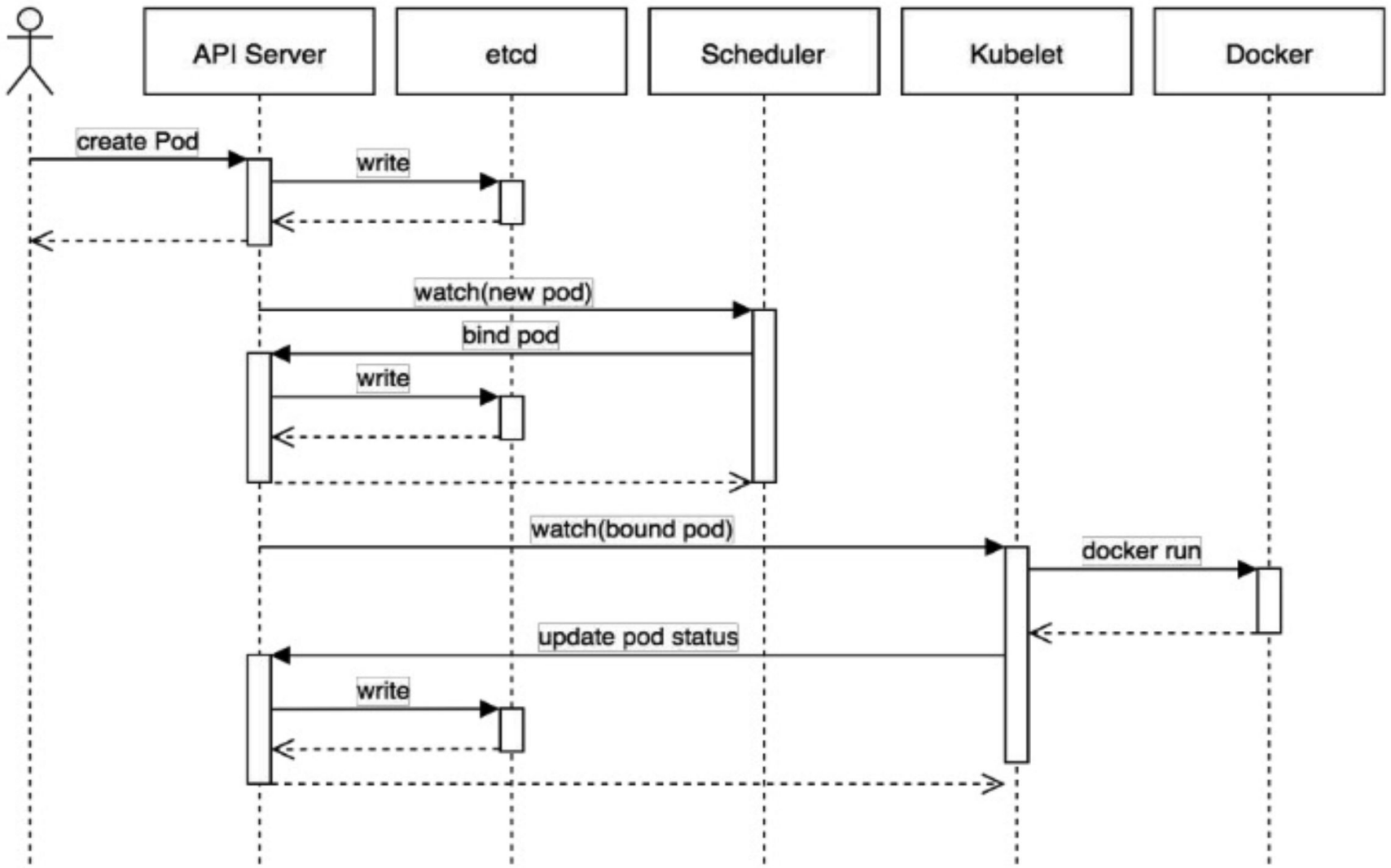
Check that the installation is successful. Starting for now all command will be the same on each platform, as a such we will keep providing a single command. We will mark with a blue book the command ( ) and a green book ( ) to show expected result.

Apache Cassandra™ with Kubernetes

1. Housekeeping and Quizz
2. Cassandra Basics
3. Kubernetes Operators
4. Cass Operator in Deep
5. Grafana | Prometheus
6. Resources

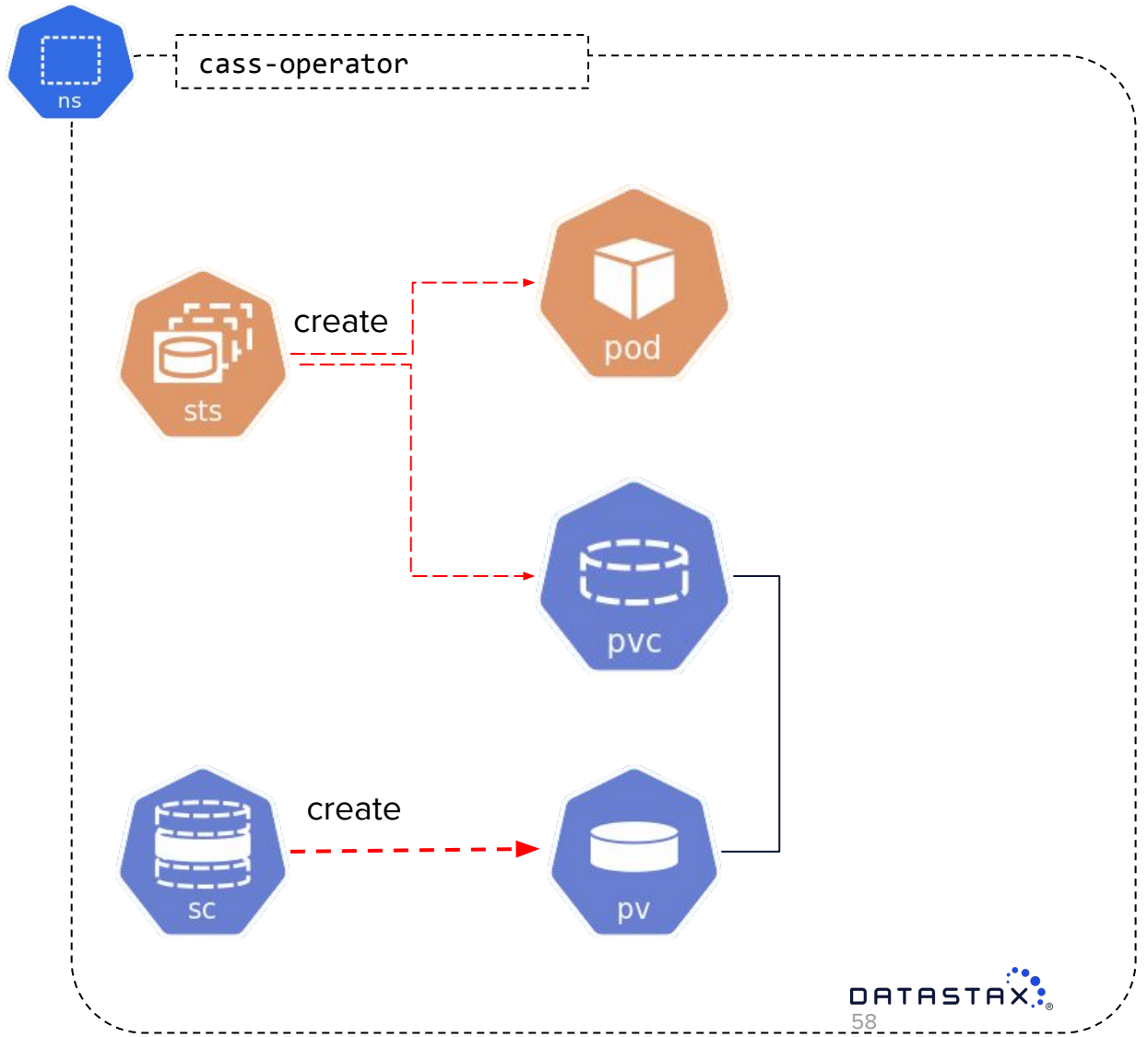
DATASTAX

Pod Lifecycle



K8s Primitives : StatefulSet

StatefulSet: StatefulSet represents a set of pods with consistent identities. Identities are defined as: network, storage.



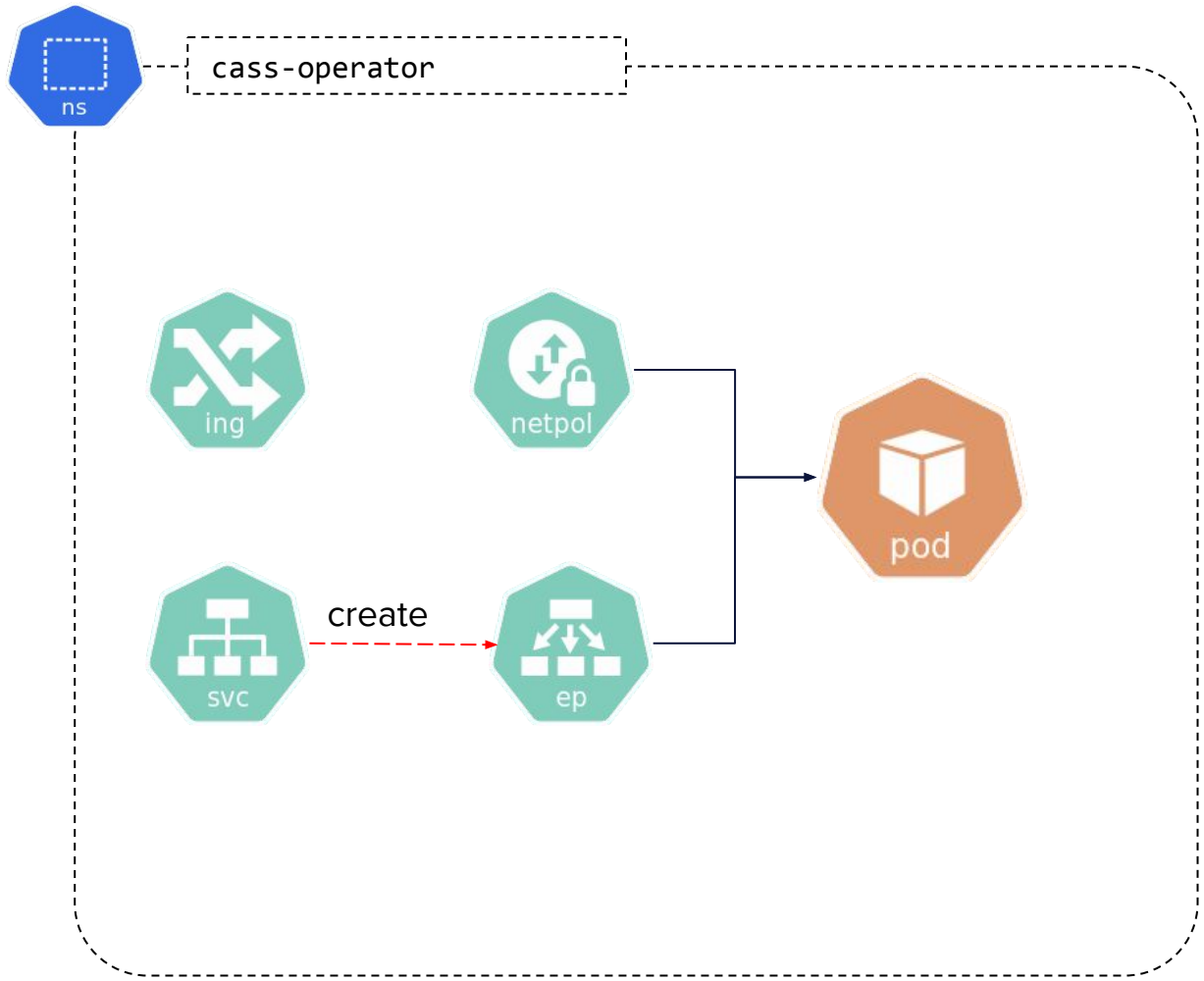
K8s Primitives : Service

Ingress is a collection of rules that allow inbound connections to reach the endpoints defined by a backend.

Service is a **named** abstraction of software service with ports to listen on and selector to determine which pods will answer requests.

EndPoint is a collection of endpoints that implement the actual service..

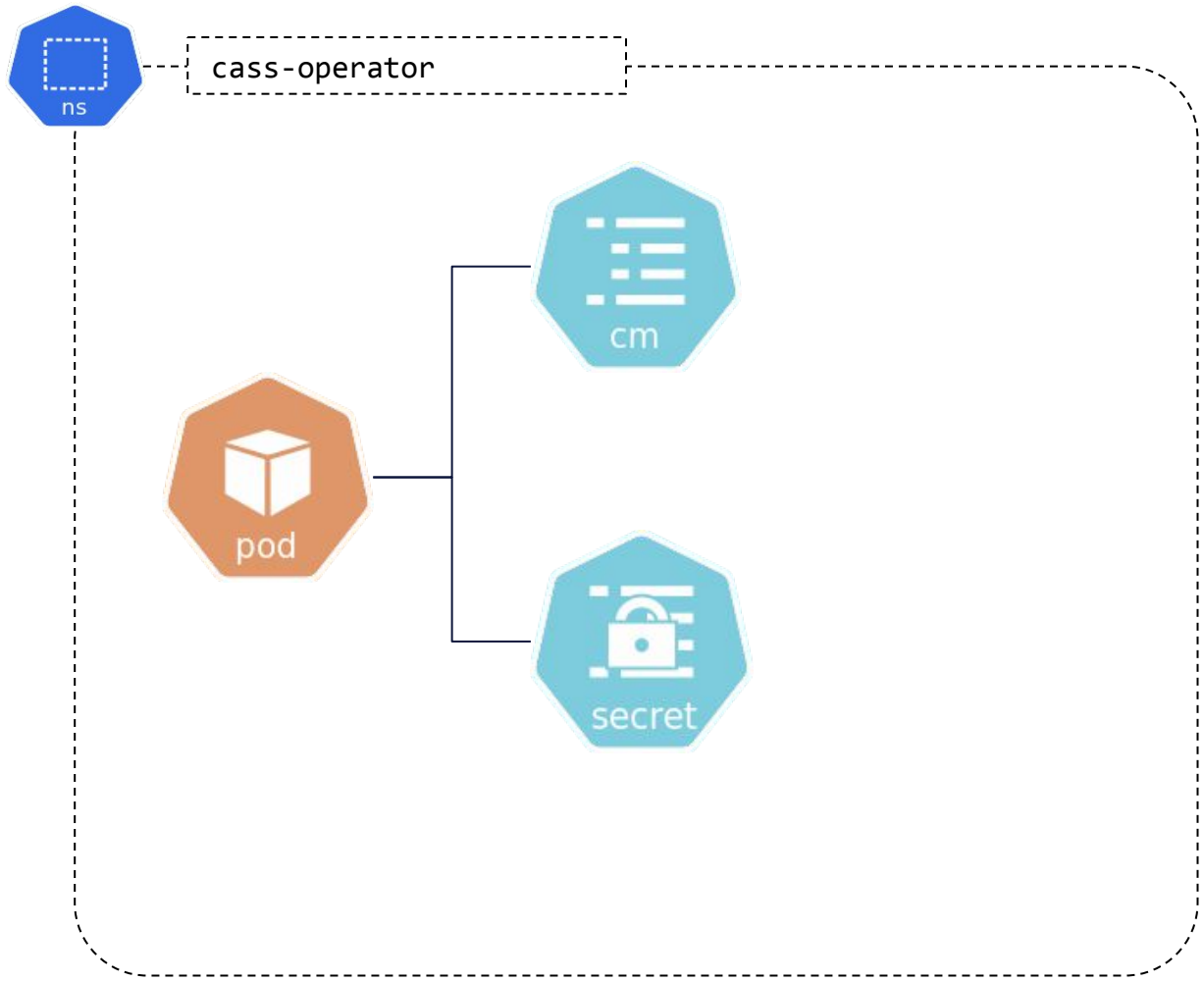
NetworkPolicy: Describes what network traffic is allowed for a set of Pods.



K8s Primitives : Configuration

ConfigMap: ConfigMap holds configuration data for pods to consume..

Secret: Secret holds secret data of a certain type..



K8s Primitives : Custom Resources

Custom Resource Definition :
Extensions of the Kubernetes API.
Customization making K8s more modular

- **Spec** declares the desired state of a resource
  - **Configuration settings** provided by the user
  - **Default values** expanded by the system
  - **Other properties** initialized by other internal components after resource creation.

- **Status** : describes the object's current, observed state.
  - Kubernetes API server provides a REST API to clients. A Kubernetes object or resource is a REST resource.
  - The status of a Kubernetes resource is typically implemented as a **REST subresource** that can only be modified by internal, system components

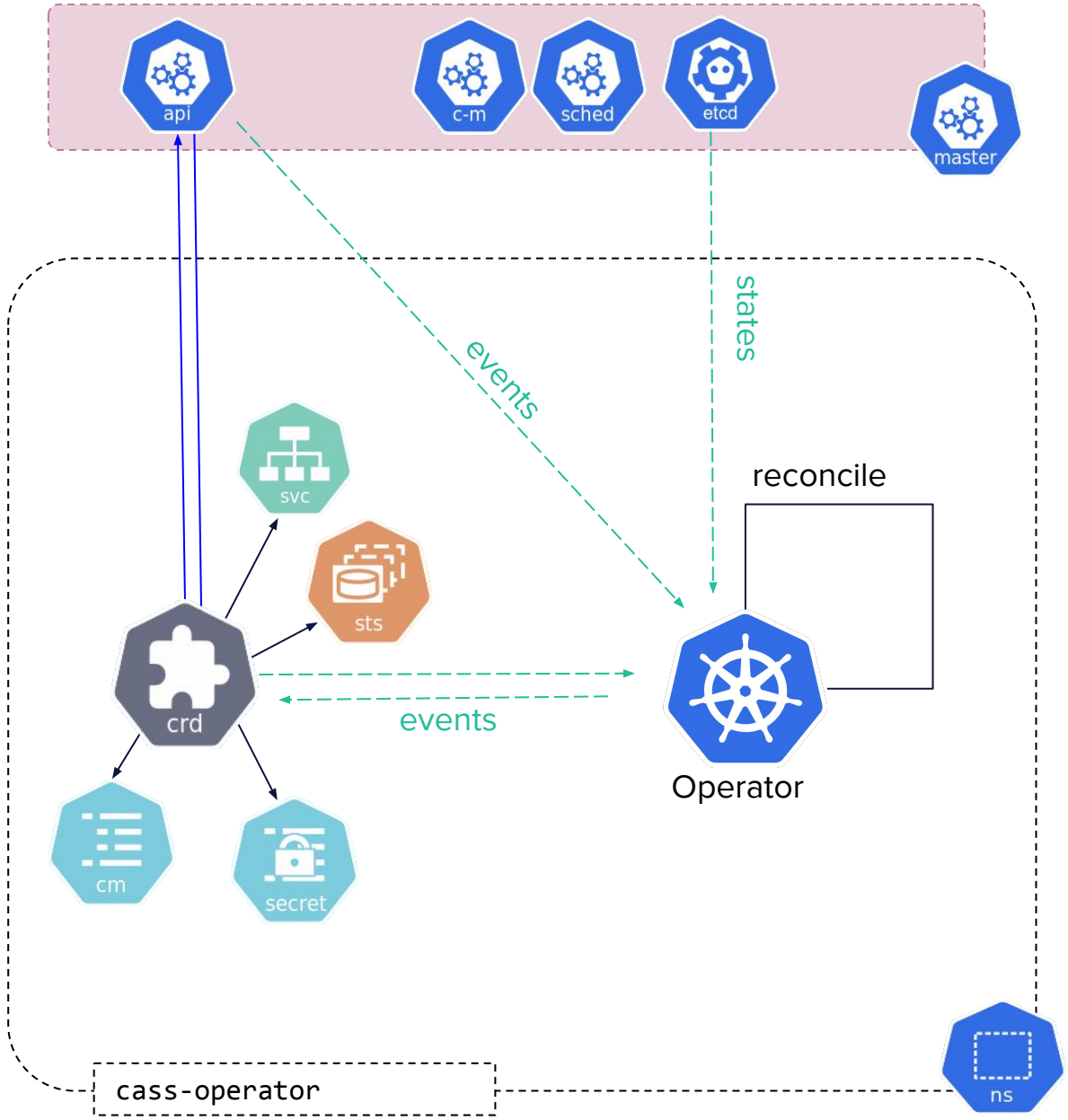
```yaml
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  name: crontabs.stable.example.com
spec:
  # group name to use for REST API: /apis/<group>/<version>
  group: stable.example.com
  # list of versions supported by this CustomResourceDefinition
  versions:
    - name: v1
      # Each version can be enabled/disabled by Served flag.
      served: true
      # One and only one version must be marked as the storage version.
      storage: true
      schema:
        openAPIV3Schema:
          type: object
          properties:
            spec:
              type: object
              properties:
                cronSpec:
                  type: string
                image:
                  type: string
                replicas:
                  type: integer
  # either Namespaced or Cluster
  scope: Namespaced
  names:
[...}
```

K8s Primitives : Operator

Building an application and driving an application on top of Kubernetes, behind Kubernetes APIs

A Kubernetes Operator helps extend the types of applications that can run on Kubernetes by allowing developers to provide additional knowledge to applications that need to maintain state.” –Jonathan S. Katz

- Reconcile CRD instances which states defined within the “**spec**” attribute.
- Listen **events** and **status evolution** to react accordingly.




menti.com
51 63 695
Mentimeter
What is the most desirable feature of your ideal workplace?
Flexible time
Social factor
Big paycheck
LIVE QUIZ
Kubernetes Quiz Time! - 6 questions
Available on the iPhone App Store
GET IT ON Google play

Apache Cassandra™ with Kubernetes

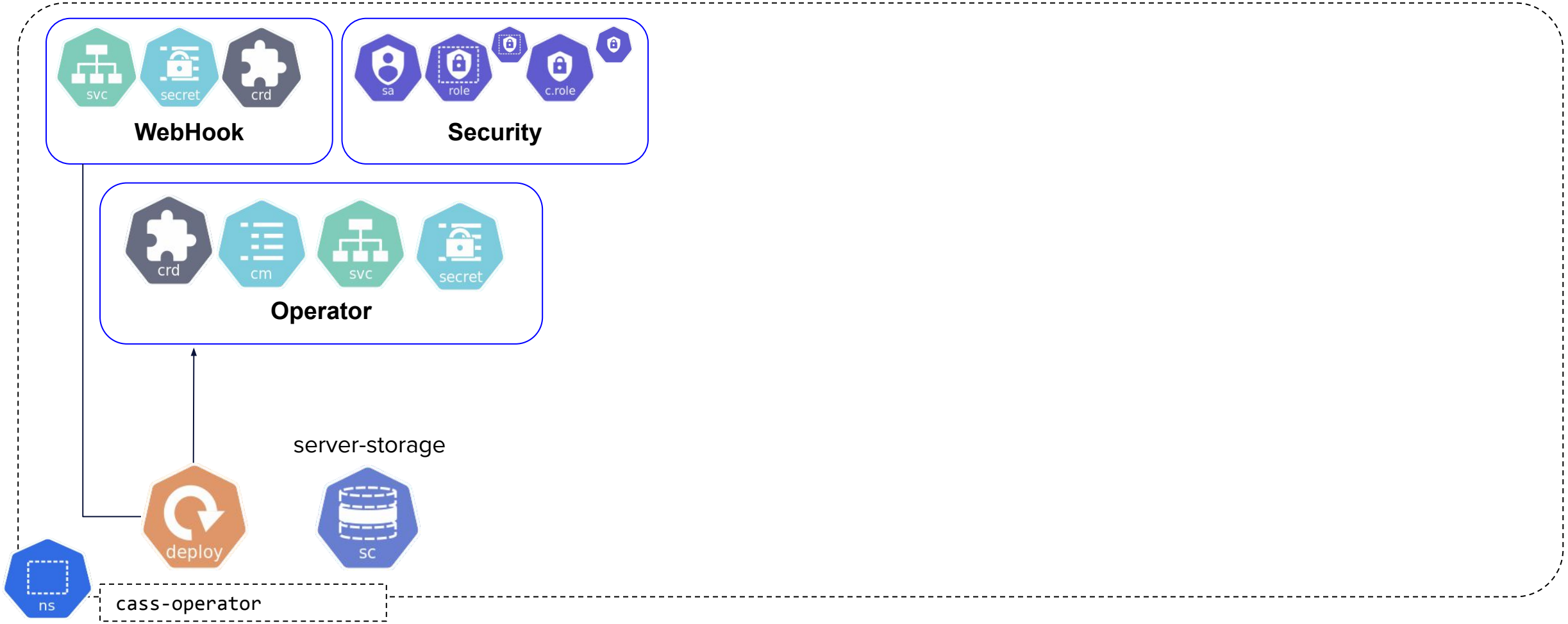
1. Housekeeping and Quizz
2. Cassandra Basics
3. Kubernetes Operators
4. Cass Operator in Deep
5. Grafana | Prometheus
6. Resources

DATASTAX

Cass Operator : Features

- Proper token ring initialization, with only one node bootstrapping at a time
- Seed node management -
  - one per rack, or three per datacenter, whichever is more
- Server configuration integrated into the CassandraDatacenter CRD
  - Rolling reboot nodes by changing the CRD
  - Store data in a rack-safe way - one replica per cloud AZ
  - Scale up racks evenly with new nodes
  - Replace dead/unrecoverable nodes
- Multi DC clusters (limited to one Kubernetes namespace)

Installing the Cass Operator Manifest



YAML :
https://github.com/DataStax-Academy/kubernetes-workshop-online/blob/master/1-cassandra/11-install-cass-operator-v1.1.yaml

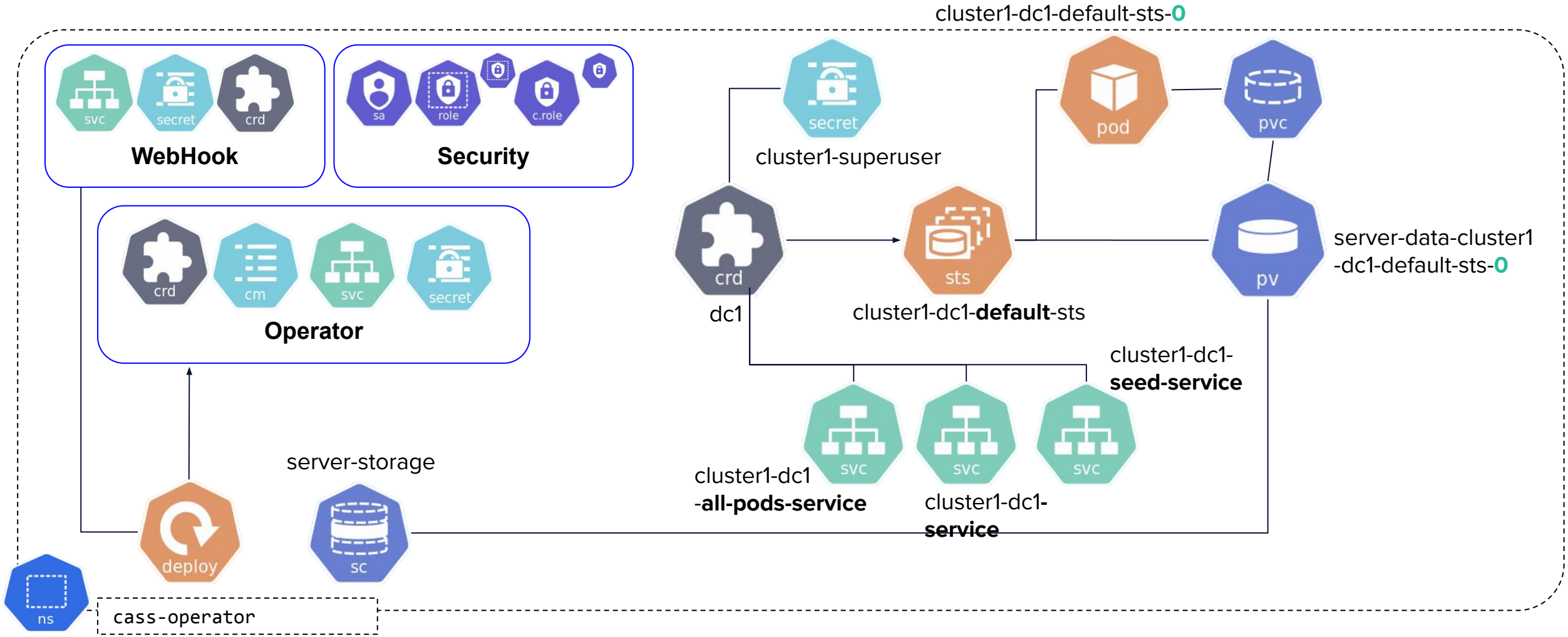
CRD CassandraDataCenter

```yaml
apiVersion: cassandra.datastax.com/v1beta1
kind: CassandraDatacenter
metadata:
  name: dc1
spec:
  clusterName: cluster1
  serverType: cassandra
  serverVersion: "3.11.6"
  managementApiAuth:
    insecure: {}
  size: 1
  storageConfig:
    cassandraDataVolumeClaimSpec:
      storageClassName: server-storage
      accessModes:
        - ReadWriteOnce
      resources:
        requests:
          storage: 5Gi
  config:
    cassandra-yaml:
      authenticator: org.apache.cassandra.auth.PasswordAuthenticator
      authorizer: org.apache.cassandra.auth.CassandraAuthorizer
      role_manager: org.apache.cassandra.auth.CassandraRoleManager
    jvm-options:
      initial_heap_size: "800M"
      max_heap_size: "800M"
```

```yaml
apiVersion: cassandra.datastax.com/v1beta1
kind: CassandraDatacenter
metadata:
  name: multi-rack
spec:
  clusterName: multi-rack
  serverType: cassandra
  serverVersion: 3.11.6
  managementApiAuth:
    insecure: {}
  size: 9
  racks:
  - name: us-east1-b
    zone: us-east1-b
  - name: us-east1-c
    zone: us-east1-c
  - name: us-east1-d
    zone: us-east1-d
  storageConfig:
    cassandraDataVolumeClaimSpec:
      storageClassName: standard
      accessModes:
      - ReadWriteOnce
      resources:
        requests:
          storage: 5Gi
```

https://github.com/DataStax-Academy/kubernetes-workshop-online/blob/master/1-cassandra/12-cassandra-cluster-1nodes.yaml

Creating DataCenter dc1



YAML :
https://github.com/DataStax-Academy/kubernetes-workshop-online/blob/master/1-cassandra/11-install-cass-operator-v1.1.yaml

DATASTAX

Our Pods

Cassandra Management API Service

https://github.com/datastax/management-api-for-apache-cassandra

Management API for Apache Cassandra 0.1 OAS3

https://raw.githubusercontent.com/datastax/management-api-for-apache-cassandra/master/management-api-server/doc/openapi.json

This is a Restful service for operating Apache Cassandra. You can find out more about the Management API on Github

Apache 2.0

default

POST	`/api/v0/ops/auth/role`	Creates a new user role
GET	`/api/v0/probes/liveness`	Indicates whether this service is running
GET	`/api/v0/probes/readiness`	Indicates whether the Cassandra service is ready to service requests
GET	`/api/v0/probes/cluster`	Indicated whether the Cassandra cluster is able to achieve the specified consistency
POST	`/api/v0/ops/seeds/reload`	
POST	`/api/v0/ops/keyspace/refresh`	Load newly placed SSTables to the system without restart
POST	`/api/v0/ops/keyspace/cleanup`	Triggers the immediate cleanup of keys no longer belonging to a node. By default, clean all keyspaces
POST	`/api/v0/lifecycle/start`	
POST	`/api/v0/lifecycle/stop`	
POST	`/api/v0/lifecycle/configure`	
GET	`/api/v0/lifecycle/pid`	
GET	`/api/v0/metadata/versions/release`	Returns the Cassandra release version
GET	`/api/v0/metadata/endpoints`	Returns this nodes view of the endpoint states of nodes
POST	`/api/v0/ops/node/drain`	Drain the node (stop accepting writes and flush all tables)

Cassandra Management API SideCar

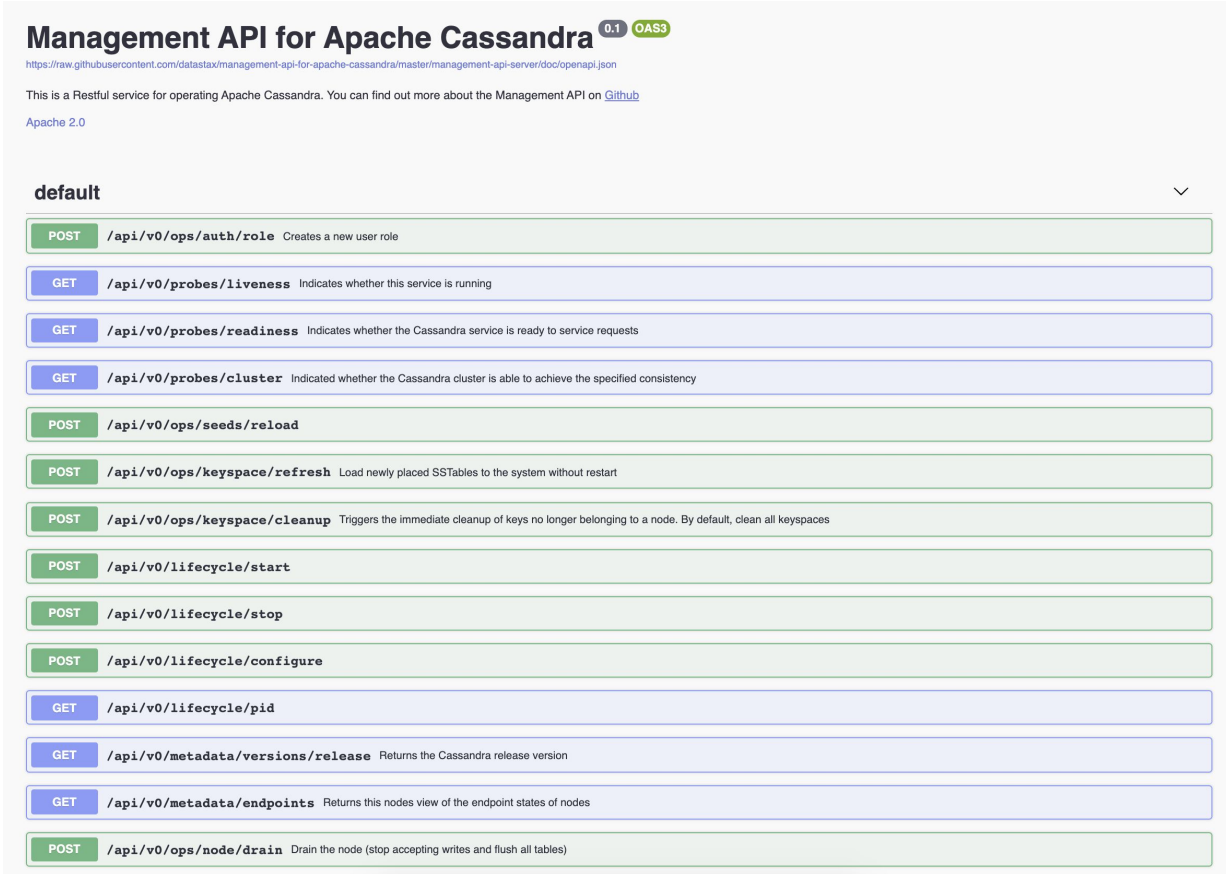
https://petstore.swagger.io/

https://raw.githubusercontent.com/datastax/management-api-for-apache-cassandra/master/management-api-server/doc/openapi.json

https://github.com/datastax/management-api-for-apache-cassandra

The Management API is a sidecar service layer that attempts to build a well supported set of operational actions on Cassandra® nodes that can be administered centrally. It currently works with official Apache Cassandra® 3.11.x an 4.0 via a drop in java agent.

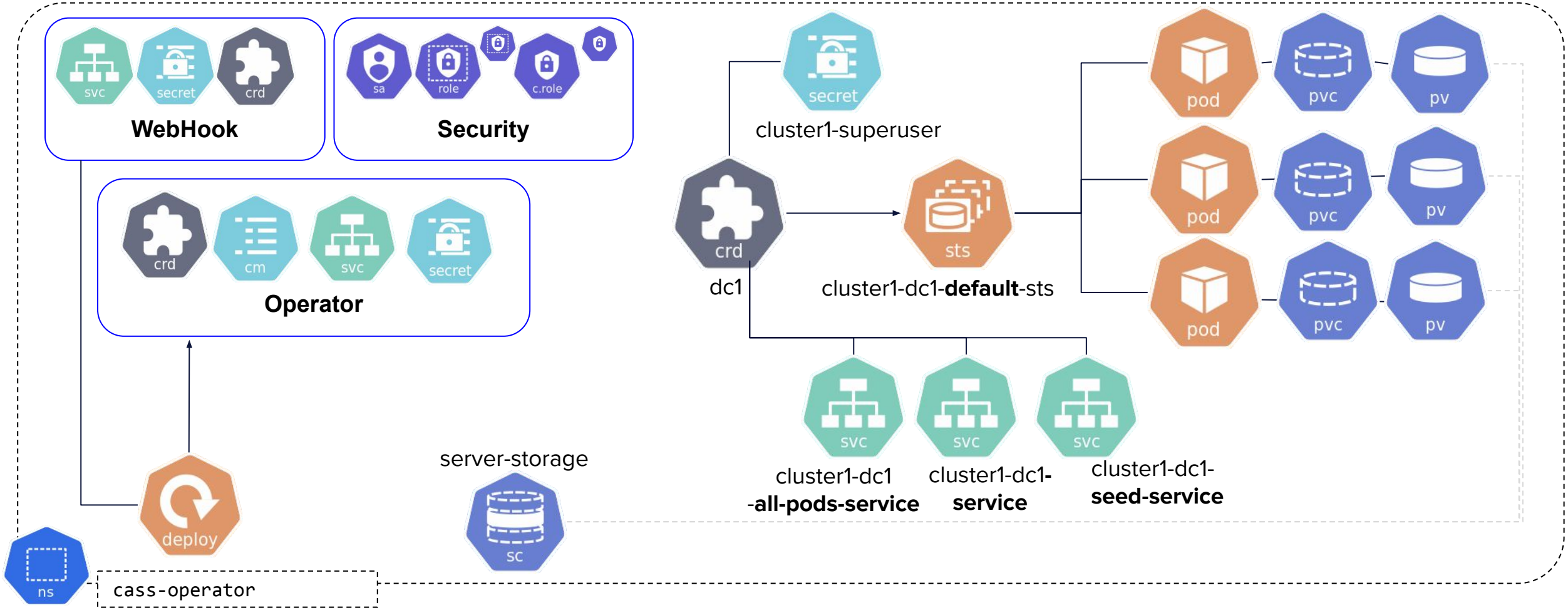
- Lifecycle Management
  - Start Node
  - Stop Node
- Configuration Management (alpha)
  - Change YAML
  - Change jvm-opts
- Health Checks
  - Kubernetes liveness/readiness checks
  - Consistency level checks
- Per node actions
  - All nodetool commands



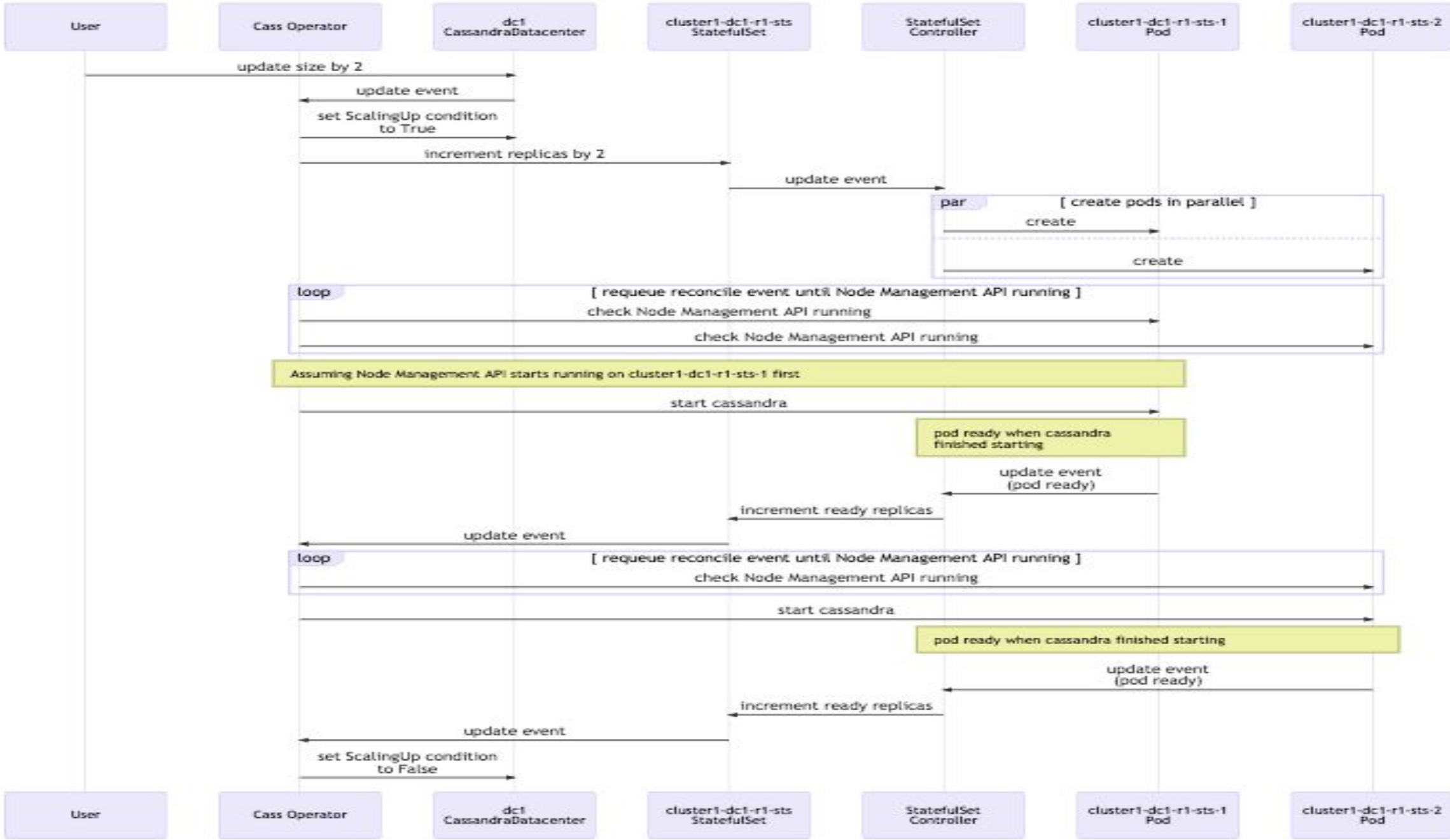
DATASTAX

Scale up DataCenter dc1

cluster1-dc1-default-sts-0
cluster1-dc1-default-sts-1
cluster1-dc1-default-sts-2



DATASTAX


User
Cass Operator
dc1
CassandraDatacenter
cluster1-dc1-r1-sts
StatefulSet
StatefulSet
Controller
cluster1-dc1-r1-sts-1
Pod
cluster1-dc1-r1-sts-2
Pod
update size by 2
update event
set ScalingUp condition
to True
increment replicas by 2
update event
par
[create pods in parallel]
create
create
loop
[requeue reconcile event until Node Management API running]
check Node Management API running
check Node Management API running
Assuming Node Management API starts running on cluster1-dc1-r1-sts-1 first
start cassandra
pod ready when cassandra
finished starting
update event
(pod ready)
increment ready replicas
update event
loop
[requeue reconcile event until Node Management API running]
check Node Management API running
start cassandra
pod ready when cassandra finished starting
update event
(pod ready)
increment ready replicas
update event
set ScalingUp condition
to False
User
Cass Operator
dc1
CassandraDatacenter
cluster1-dc1-r1-sts
StatefulSet
StatefulSet
Controller
cluster1-dc1-r1-sts-1
Pod
cluster1-dc1-r1-sts-2
Pod

Online Workshops

DATASTAX

Section #1 : Cass Operator

https://github.com/DataStax-Academy/kubecon-cassandra-workshop/tree/master/1-cassandra

2. Create a single node cluster

Apply this file via `kubectl` and watch the list of pods as the operator deploys them. Completing a deployment may take several minutes per node.

2a. Create the cluster

```
kubectl -n cass-operator apply -f ./1-cassandra/12-cassandra-cluster-1nodes.yaml
```

2b. Watch progression

```
watch kubectl -n cass-operator get pod
```

Expected output

```
NAME                              READY   STATUS    RESTARTS   AGE
cass-operator-657cb5c695-q9psl    1/1     Running   0          5m22s
cluster1-dc1-default-sts-0        1/2     Running   0          50s
```

2c. Execute the command to describe the datacenter

```
kubectl -n cass-operator describe cassdc dc1
```

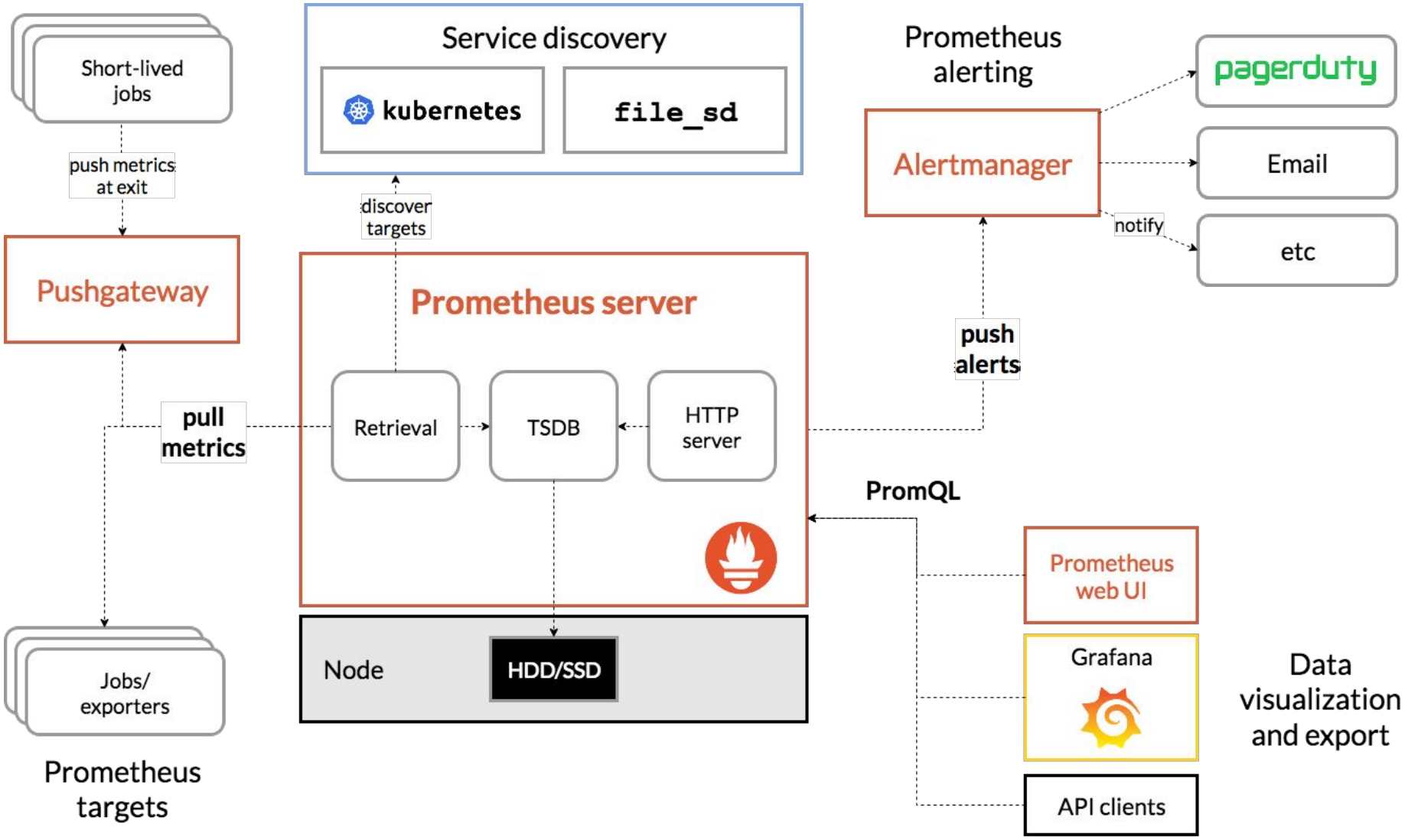
Expected output

Apache Cassandra™ with Kubernetes

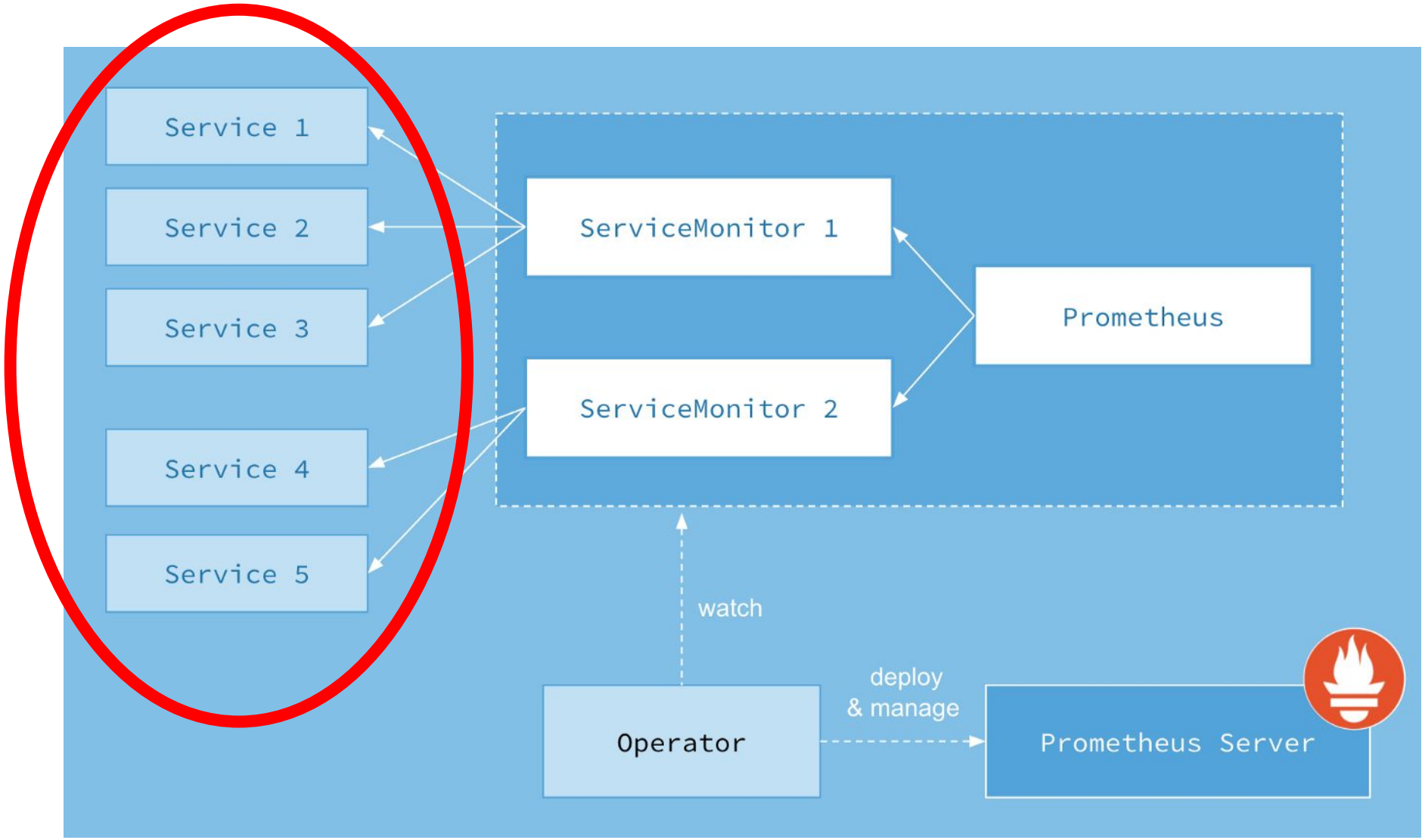
1. Housekeeping and Quizz
2. Cassandra Basics
3. Kubernetes Operators
4. Cass Operator in Deep
5. Grafana | Prometheus
6. Resources

DATASTAX

Prometheus + Grafana



Prometheus metrics collection



DSE Metrics Exporter

Reference Documentation

https://docs.datastax.com/en/dse/6.7/dse-dev/datastax_enterprise/tools/metricsCollector/mcIntroduction.html

- DSE Metrics Collector aggregates DataStax Enterprise (DSE) metrics and integrates with existing monitoring solutions to facilitate problem resolution and remediation.
- **DSE Metrics Collector is built on collectd,** a popular, well-supported, open source metric collection agent. With over 90 plugins, you can tailor the solution to collect metrics most important to your organization.
- When DSE Metrics Collector is enabled, DSE sends metrics and other structured events to DSE Metrics Collector.

`/etc/dse/collectd.conf.tmpl`

```
LoadPlugin load
LoadPlugin memory
LoadPlugin swap
LoadPlugin uptime
LoadPlugin processes
LoadPlugin tcpconns
```

MCAC Metrics Collector for Apache Cassandra

Reference Documentation

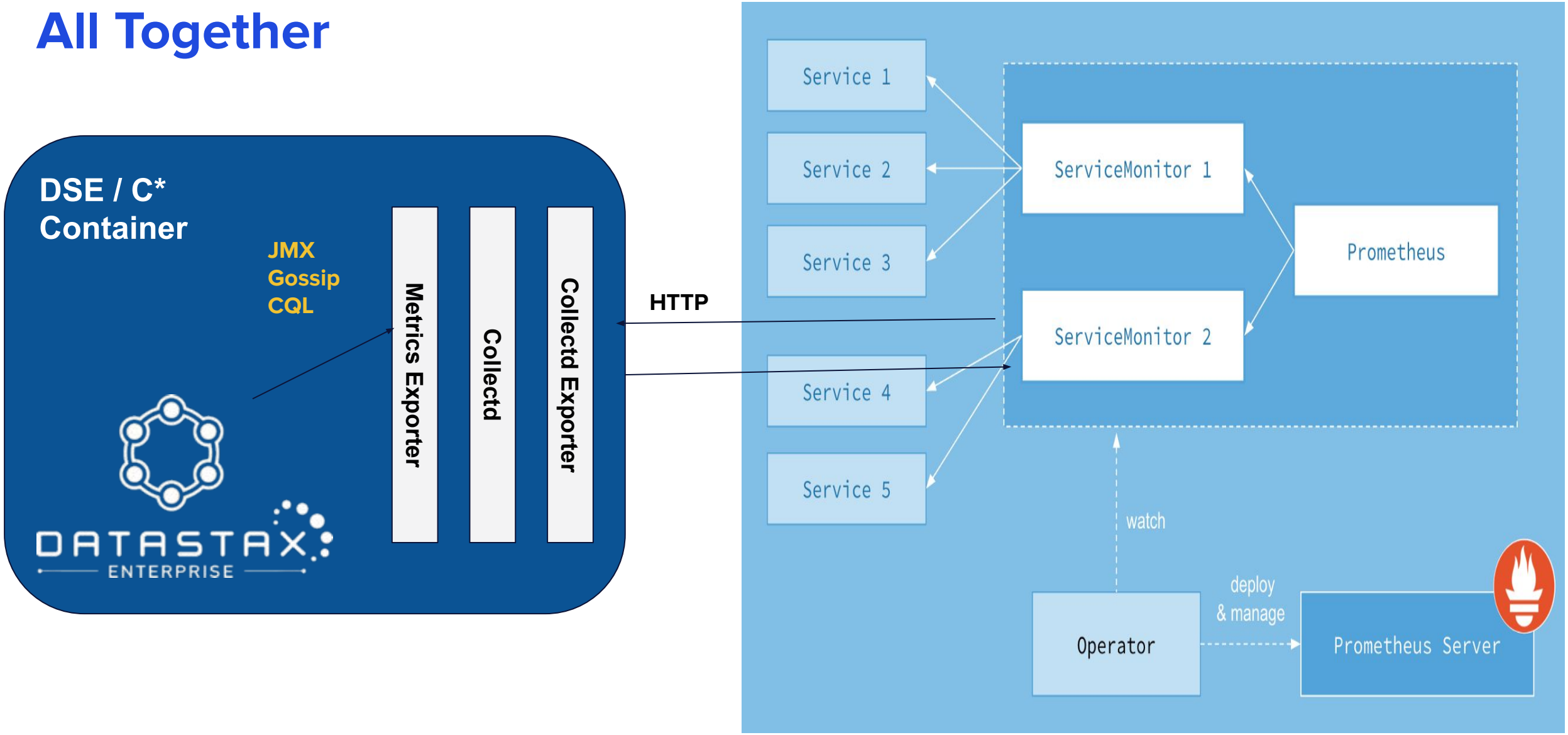
https://github.com/datastax/metric-collector-for-apache-cassandra

- MCAC Metrics Collector aggregates Apache Cassandra metrics and integrates with existing monitoring solutions to facilitate problem resolution and remediation.
- MCAC is built on **collectd**, a popular, well-supported, open source metric collection agent. With over 90 plugins, you can tailor the solution to collect metrics most important to your organization.
- Easily added to Cassandra nodes as a java agent, Apache Cassandra sends metrics and other structured events to collectd over a local unix socket.

/etc/dse/collectd.conf.tmpl

```
LoadPlugin load

LoadPlugin memory

LoadPlugin swap

LoadPlugin uptime

LoadPlugin processes

LoadPlugin tcpconns
```

All Together



Online Workshops

DATASTAX

Section #2 :
Grafana Prometheus

https://github.com/DataStax-Academy/kubecon-cassandra-workshop/tree/master/2-prometheus_grafana

Apache Cassandra™ with Kubernetes

1. Housekeeping and Quizz
2. Cassandra Basics
3. Kubernetes Operators
4. Cass Operator in Deep
5. Grafana | Prometheus
6. Resources

DATASTAX

Developer Resources

LEARN

Join academy.datastax.com
Free online courses - Cassandra certifications

ASK/SHARE

Join community.datastax.com
Ask/answer community user questions - share your expertise

SUBSCRIBE !

Follow us @DataStaxDevs
We are on Twitter - Youtube - Twitch!

Slides and code for this course are available at
https://github.com/DataStax-Academy/kubecon-cassandra-workshop

REVIEW

Accelerate
https://www.datastax.com/accelerate/why-tomorrows-cassandra-deployments-will-be-on-kubernetes

DATASTAX

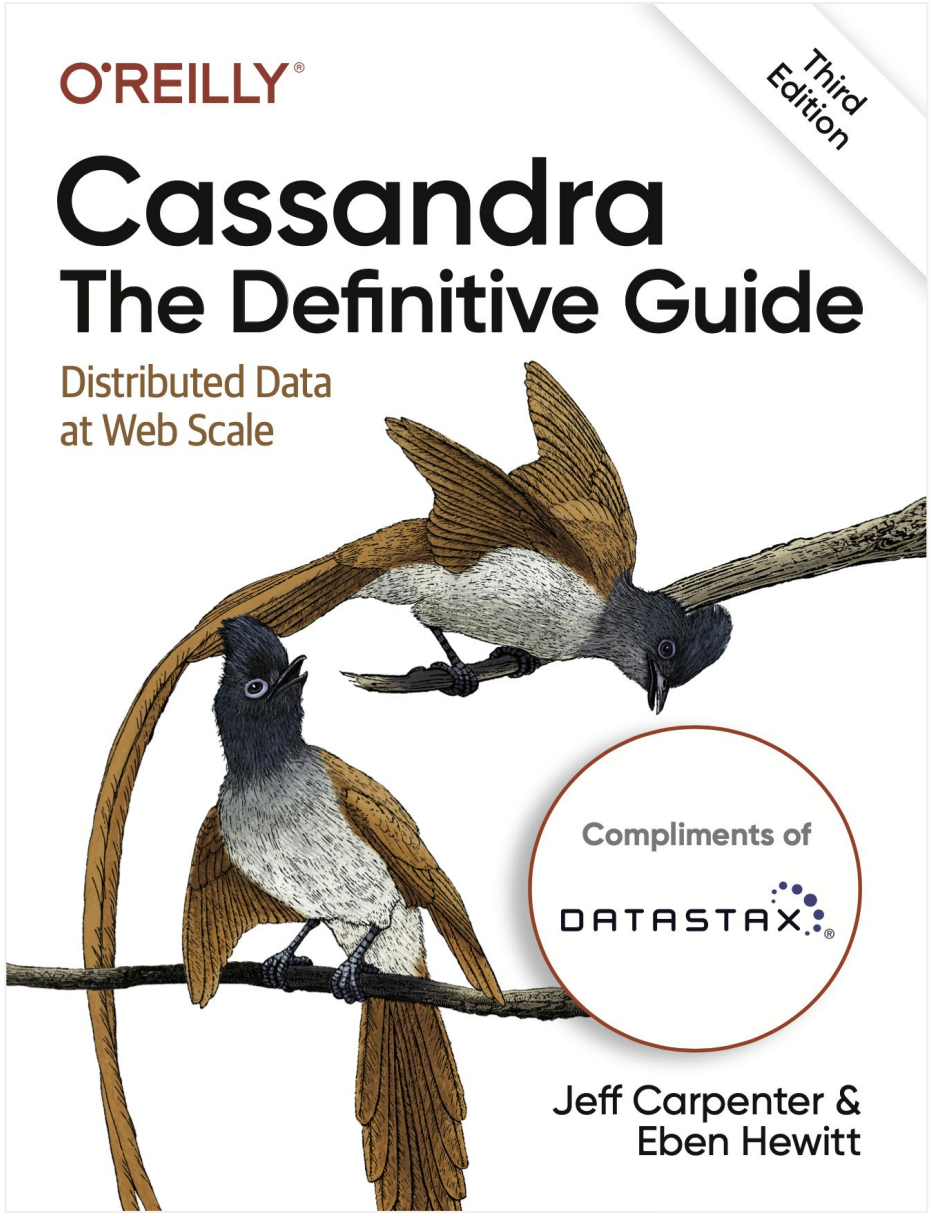
Training Courses at DataStax Academy

- Free self-paced DSE 6 courses
  - DS201: Foundations of Apache Cassandra™
  - DS210: Operations with Apache Cassandra™
  - DS220: Practical Application Data Modeling with Apache Cassandra™
  - DS330: DataStax Enterprise 6 Graph
  - DS332: DataStax Enterprise 6 Graph Analytics
  - **And MUCH MORE!**

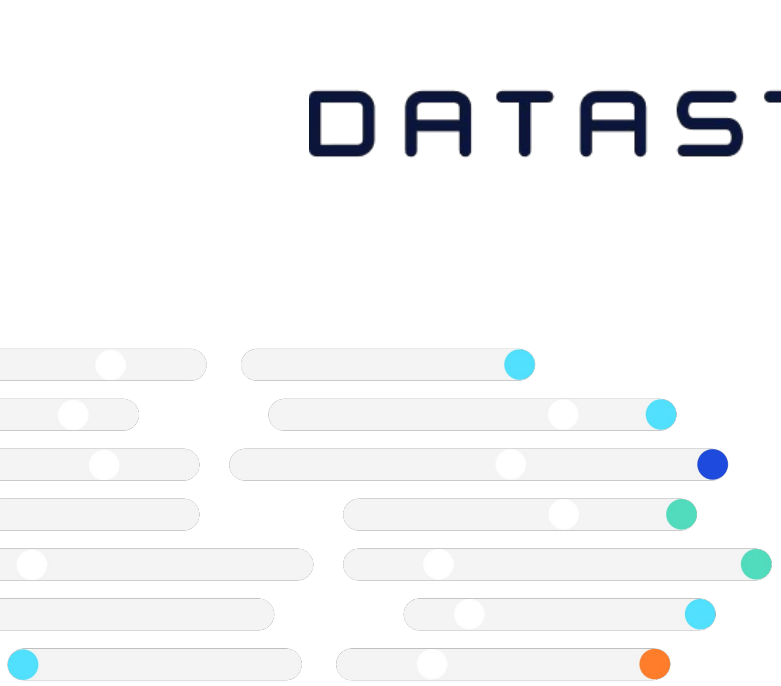
DATASTAX

Cassandra The Definitive Guide

https://www.datastax.com/resources/ebook/oreilly-cassandra-definitive-guide




menti.com
51 63 695
Mentimeter
What is the most desirable feature of your ideal workplace?
Flexible time
Social factor
Big paycheck
Available on the iPhone
App Store
GET IT ON
Google play



Thank you

SURVEY