Apache CASSANDRA

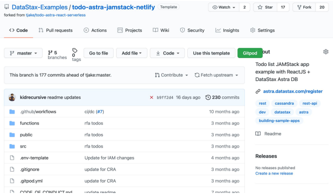
Apache Cassandra® Hands-On Workshop

Sponsored by DataStax

Hands-On Housekeeping

Nothing to install

Source code + exercises + slides



DataStax-Examples / todo-astra-jamstack-netlify Template

forked from tjake/todo-astra-react-serverless

This branch is 177 commits ahead of tjake:master.

kidrecursive readme updates ✕ b9ff2d4 16 days ago 230 commits

.github/workflows	ci/dc (#7)	10 months ago
functions	rfa todos	3 months ago
public	rfa todos	3 months ago
src	rfa todos	3 months ago
.env-template	Update for IAM changes	4 months ago
.gitignore	update for CRA	3 months ago
.gitpod.yml	update for CRA	3 months ago
CODE_OF_CONDUCT.md	update readme	7 months ago

out

Todo list JAMStack app example with ReactJS + DataStax Astra DB

astra.datastax.com/register

rest cassandra rest-api dev datastax astra building-sample-apps

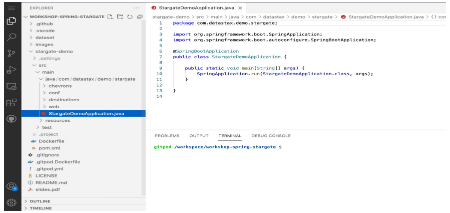
Releases

No releases published

Create a new release

GitHub

IDE



```java
package com.datastax.demo.stargate;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class StargateDemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(StargateDemoApplication.class, args);
    }

}
```

```
gitpod /workspace/workshop-spring-stargate $
```

Gitpod

Database + Api + Streaming

DataStax Astra DB

Cassandra Days

01

Getting started with Apache Cassandra®

02

Hands-on lab 1:
Create a database

03

Data definition and manipulation with CQL

04

Hands-on lab 2:
Create, populate and query

05

Application development in Java and Python

06

Hands-on lab 3:
Create an app with C* driver

The most scalable NoSQL database

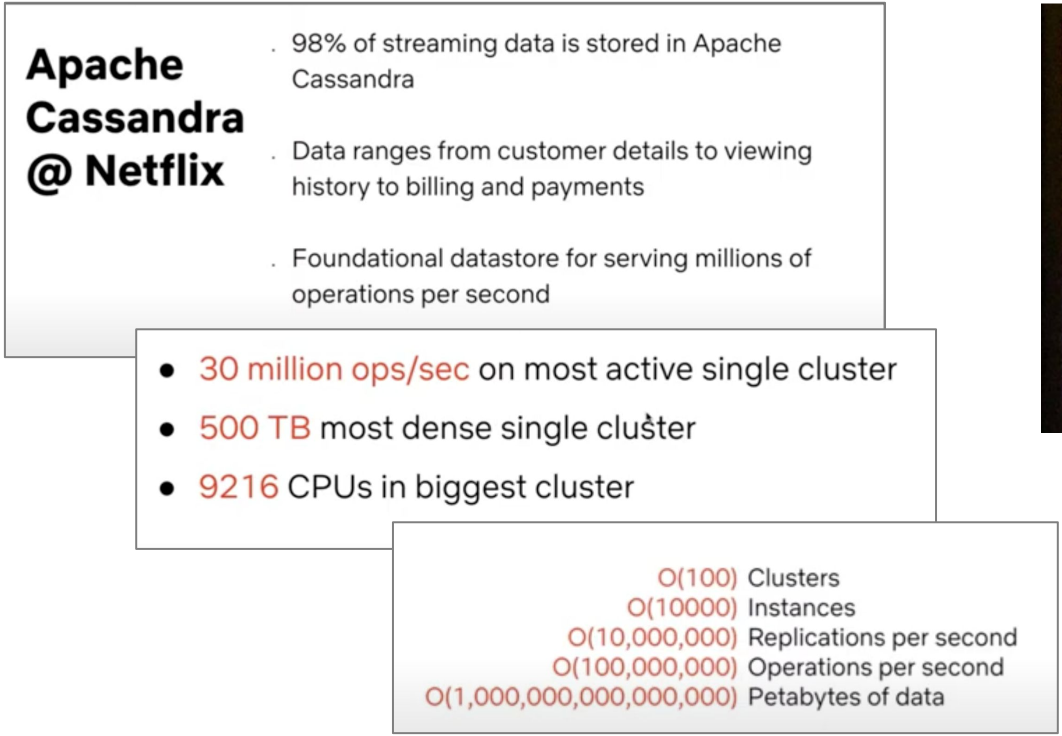
Apache Cassandra

Apache Cassandra

NoSQL Distributed Decentralised Database Management System

- Distributed, real-time, OLTP, NoSQL
- Linear Scalability
- High Availability
- Geographical Distribution
- Platform Agnostic
- Vendor Independent

Cassandra Biggest Users (and Developers)



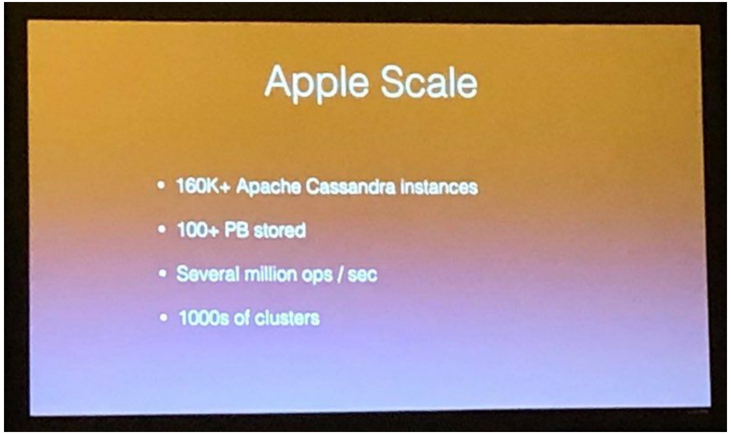
Apache Cassandra @ Netflix

- 98% of streaming data is stored in Apache Cassandra
- Data ranges from customer details to viewing history to billing and payments
- Foundational datastore for serving millions of operations per second

- 30 million ops/sec on most active single cluster
- 500 TB most dense single cluster
- 9216 CPUs in biggest cluster

O(100)	Clusters
O(10000)	Instances
O(10,000,000)	Replications per second
O(100,000,000)	Operations per second
O(1,000,000,000,000,000)	Petabytes of data

dtsx.io/cassandra-at-netflix

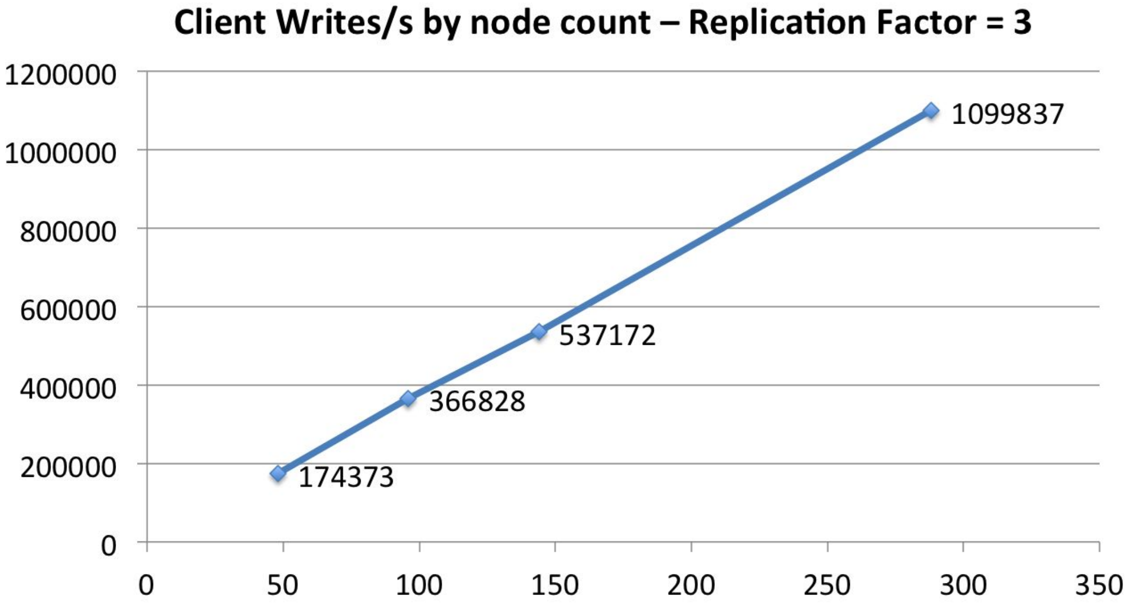


Apple Scale

- 160K+ Apache Cassandra instances
- 100+ PB stored
- Several million ops / sec
- 1000s of clusters

And many others...

Linear Scalability

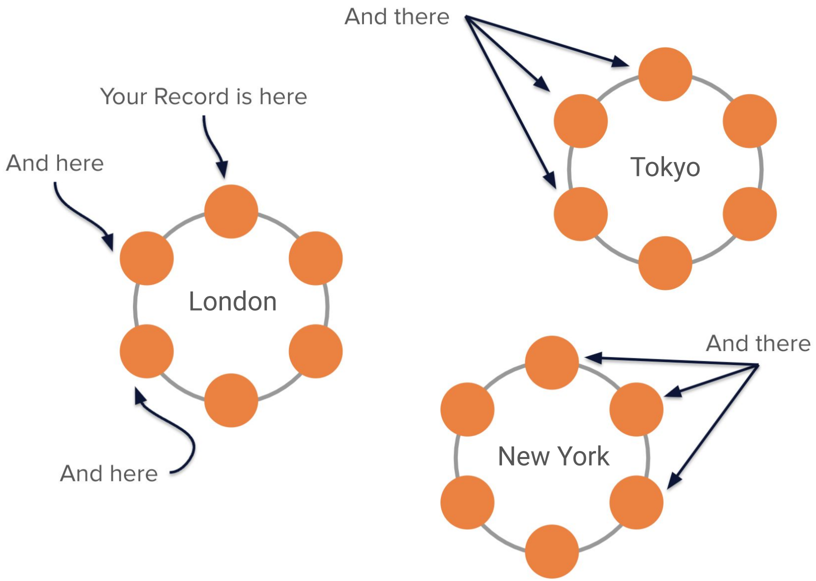


NETFLIX

High Availability

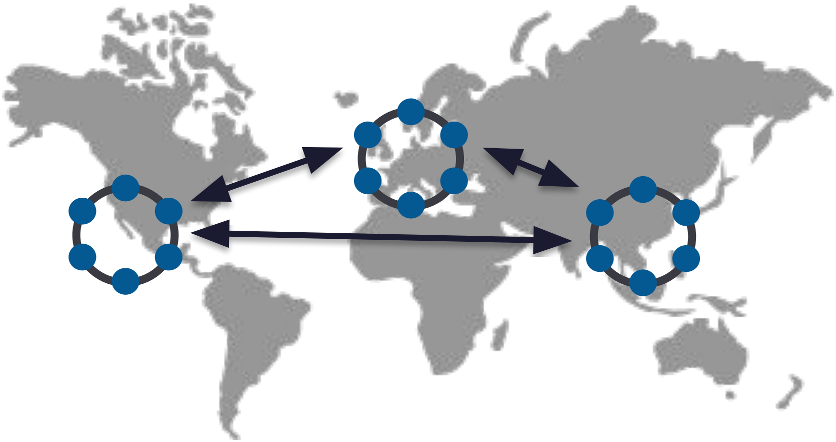
Replication, Decentralisation, and Topology-Aware Placement Strategy take care of possible downtimes:

- Multiple Live Replicas
- No Single Point of Failure
- Network topology-aware data placement
- Client-side Smart Reconnection and Strong Retry Mechanism



Geographical Distribution

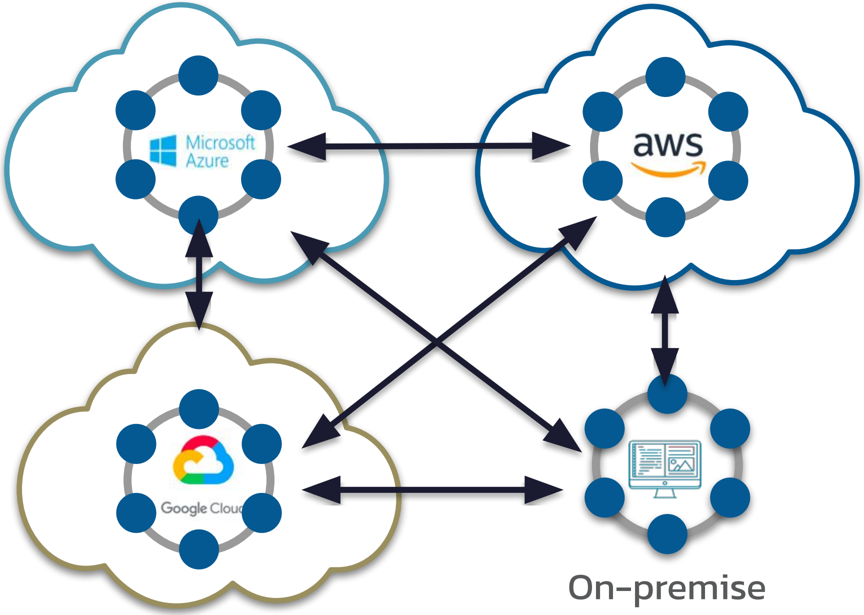
Cassandra's trademark is multi-datacenter deployments, granting you an exceptional capability for disaster tolerance while keeping your data close to your clients - worldwide.



All DCs are active (available for both writes and reads)!

Platform Agnostic

Apache Cassandra is **not bound to any platform** or service provider, helping you build hybrid-cloud and multi-cloud solutions with ease.



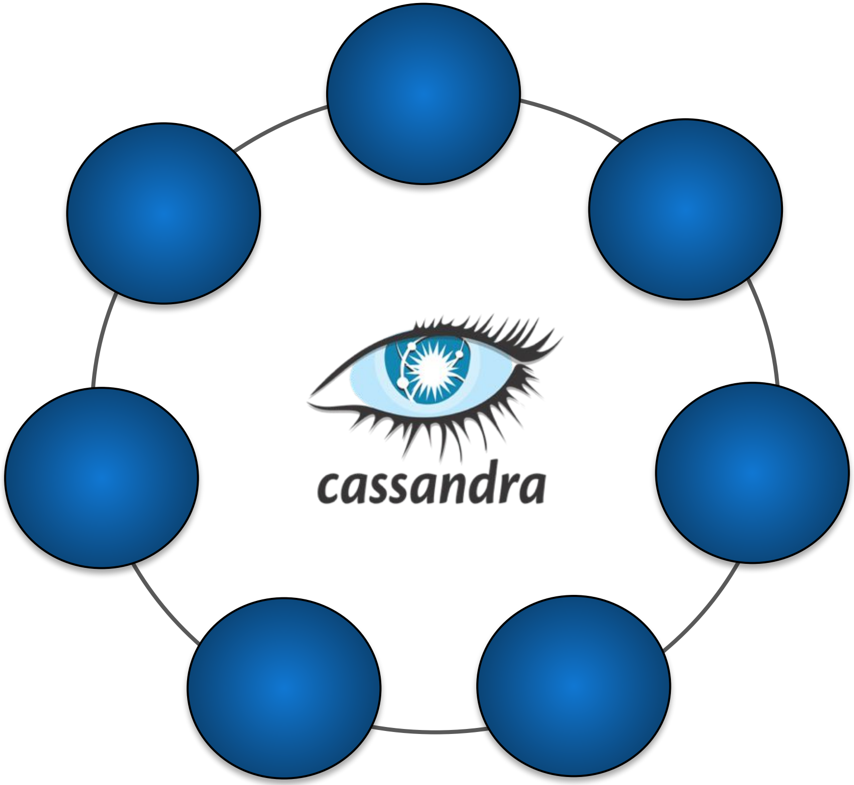
Vendor Independent

Cassandra doesn’t belong to any of commercial vendors but controlled by a non-profit Open Source **Apache Software Foundation**, already familiar to you by *Hadoop*, *Spark*, *Kafka*, *Zookeeper, Maven* and many other projects.

THE APACHE®
SOFTWARE FOUNDATION
20TH ANNIVERSARY

Data distribution, replication and consistency

Partition key and partitions



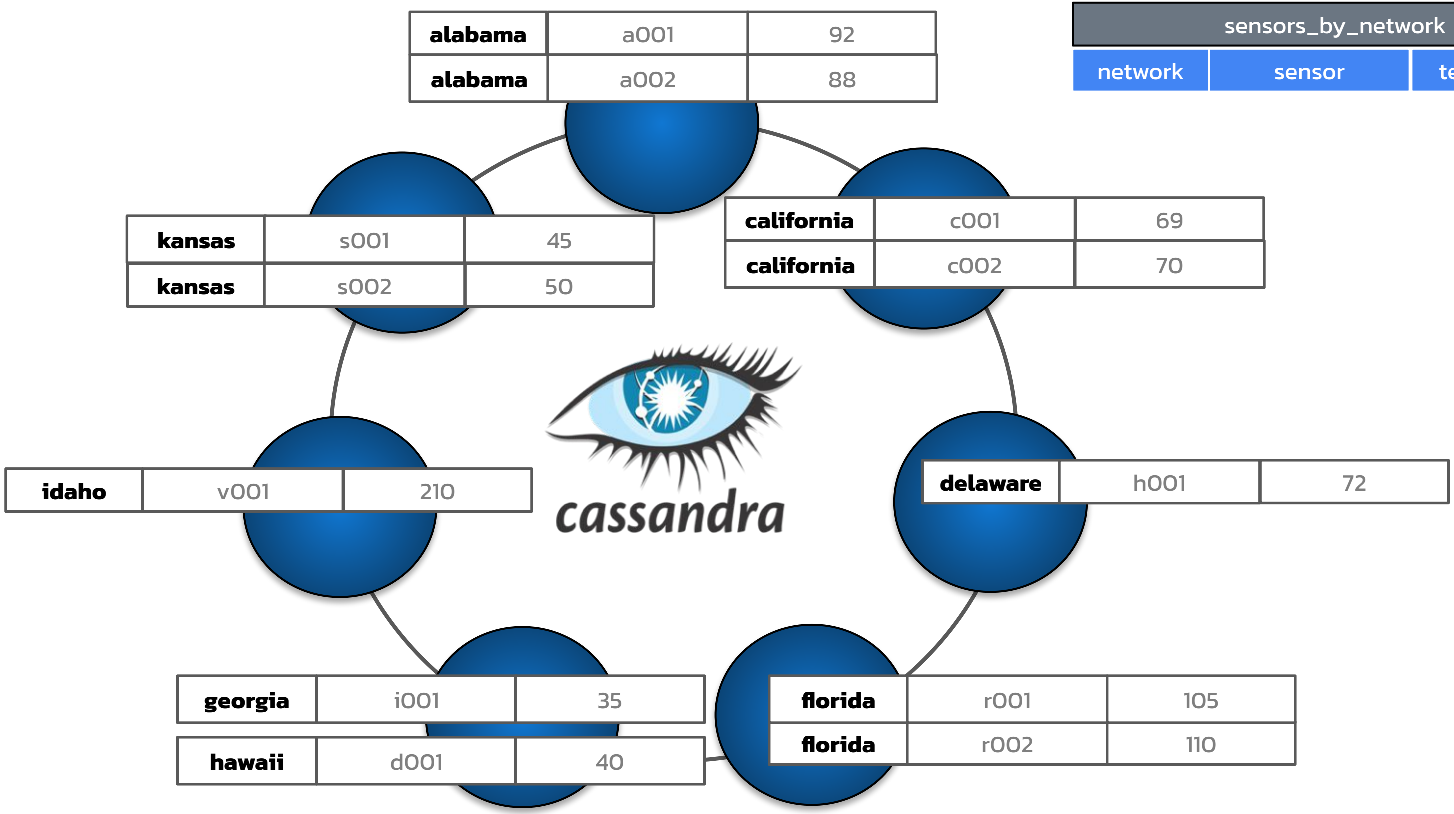
sensors_by_network

network	sensor	temperature
alabama	a001	92
alabama	a002	88
california	c001	210
delaware	d001	45
delaware	d002	50
florida	f001	72
georgia	g001	69
georgia	g002	70
hawaii	h001	40
idaho	i001	105
idaho	i002	110
kansas	k001	35

Partition Key

Data distribution

sensors_by_network		
network	sensor	temperature



alabama	a001	92
alabama	a002	88

kansas	s001	45
kansas	s002	50

california	c001	69
california	c002	70

idaho	v001	210

delaware	h001	72

georgia	i001	35

hawaii	d001	40

florida	r001	105
florida	r002	110

Partition key definition in CQL

Table

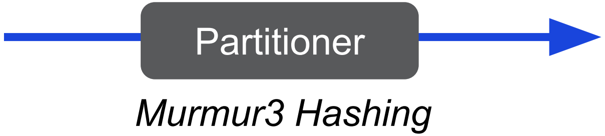
```
CREATE TABLE sensor_data.sensors_by_network (
    network     text,
    sensor      text,
    temperature integer,
    PRIMARY KEY ((network), sensor)
);
```

Partition key

Internals: Partitioning and Token Ranges

network	sensor	Value
alabama	f001	92
alabama	f002	88
idaho	s001	45
idaho	s002	50
hawaii	r001	105
hawaii	r002	110

Partition Keys

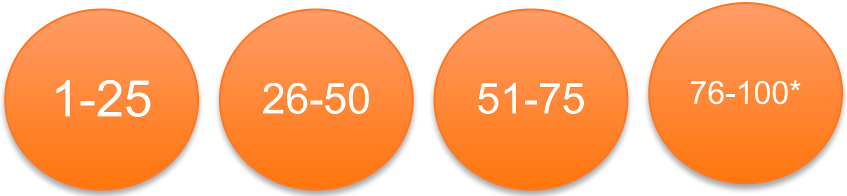


Murmur3 Hashing

Network	Sensor	Value
59	f001	92
59	f002	88
12	s001	45
12	s002	50
45	r001	105
45	r002	110

Tokens

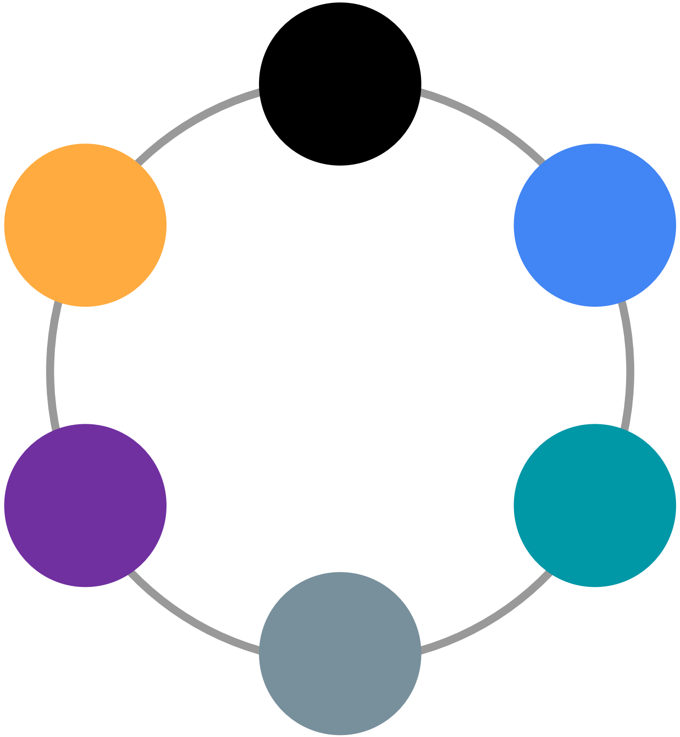
Cassandra Nodes



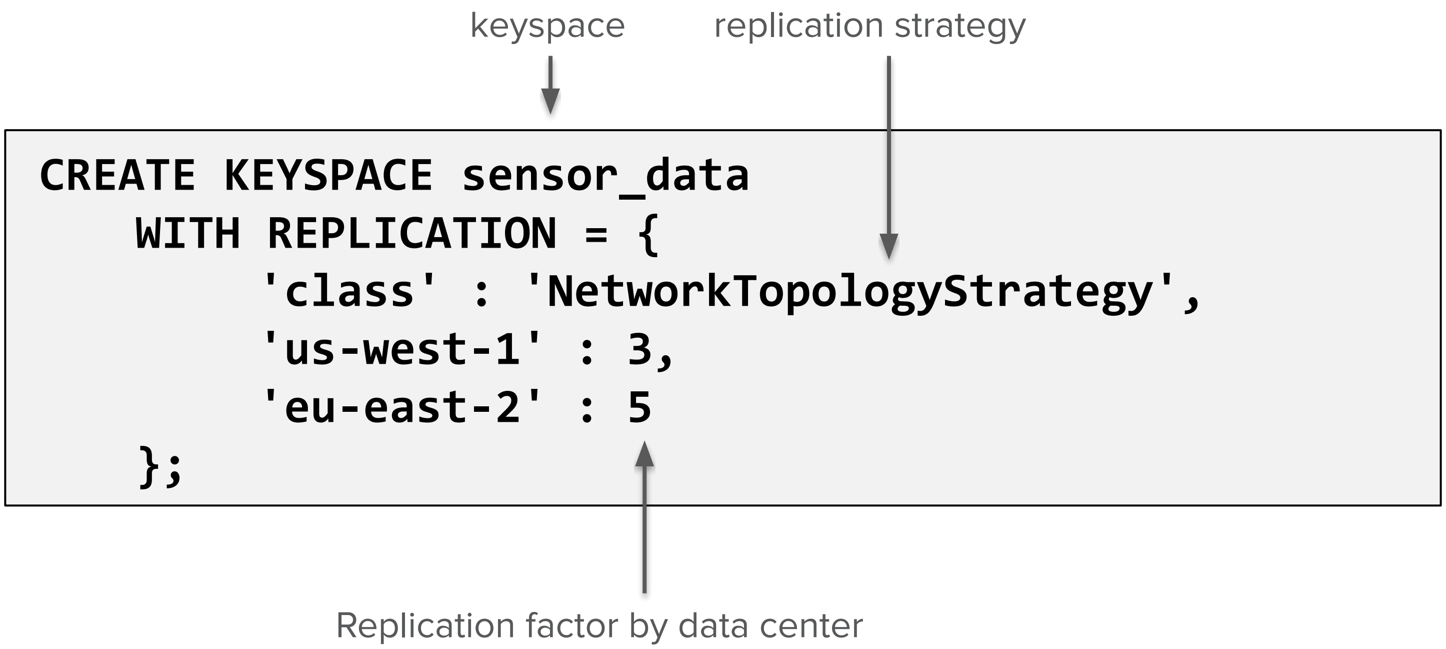
Replication Factor

RF = ?

Replication Factor means the number of nodes used to store each partition



Replication is defined per keyspace



keyspace

replication strategy

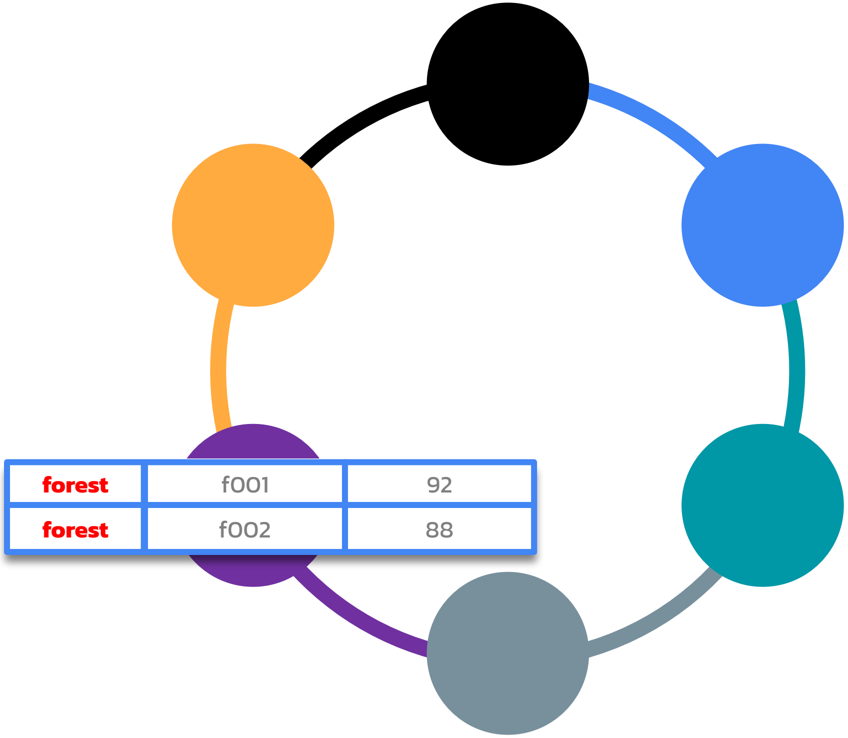
```
CREATE KEYSPACE sensor_data
    WITH REPLICATION = {
          'class' : 'NetworkTopologyStrategy',
          'us-west-1' : 3,
          'eu-east-2' : 5
    };
```

Replication factor by data center

RF = 1

RF = 1

Replication Factor 1 means that every partition is stored on 1 node

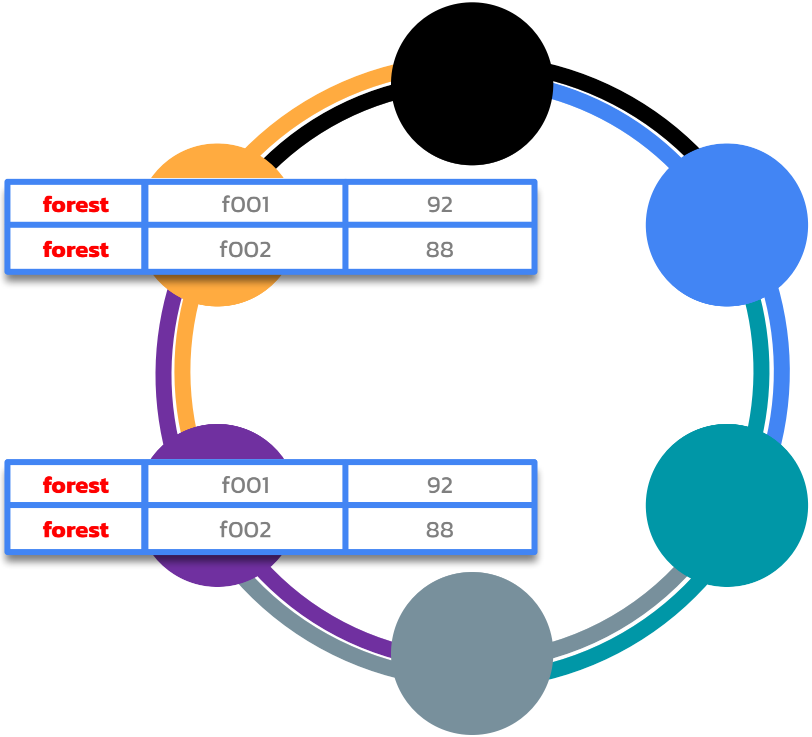


forest	f001	92
forest	f002	88

RF = 2

RF = 2

Replication Factor 2 means that every partition is stored on 2 nodes



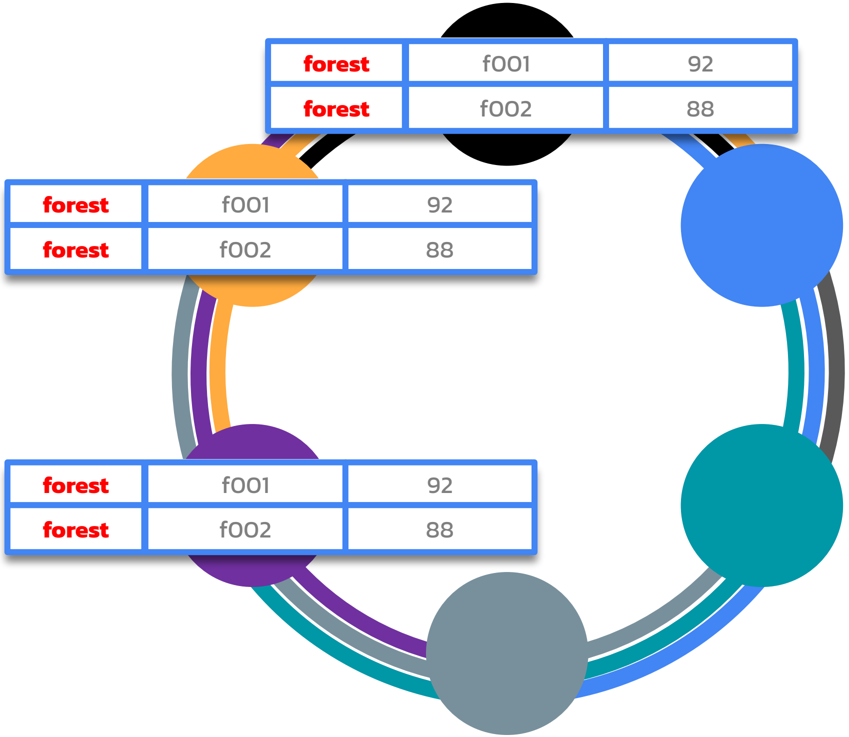
forest	f001	92
forest	f002	88

forest	f001	92
forest	f002	88

RF = 3

RF = 3

Replication Factor 3 means that every partition is stored on 3 nodes



forest	f001	92
forest	f002	88

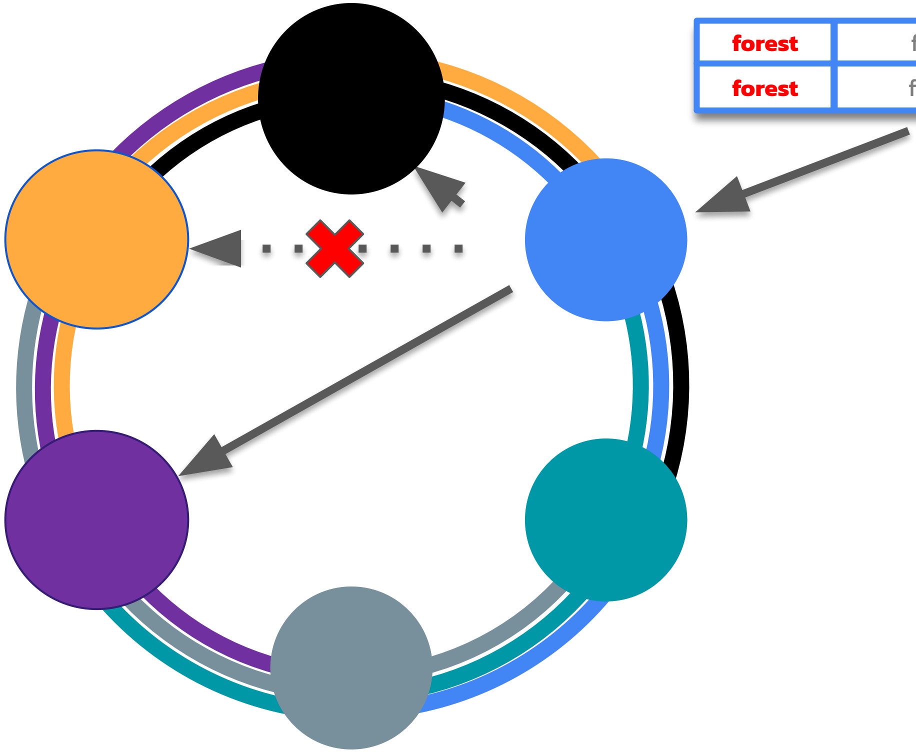
forest	f001	92
forest	f002	88

forest	f001	92
forest	f002	88

Replication, consistency and availability

But what if ...

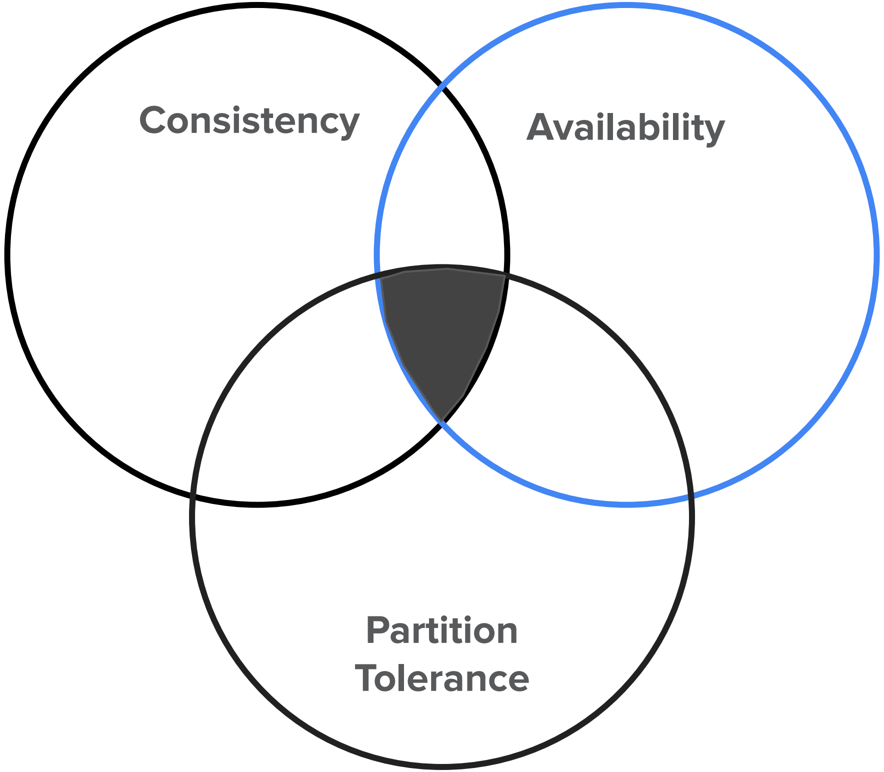
RF = 3



forest	f001	92
forest	f002	88

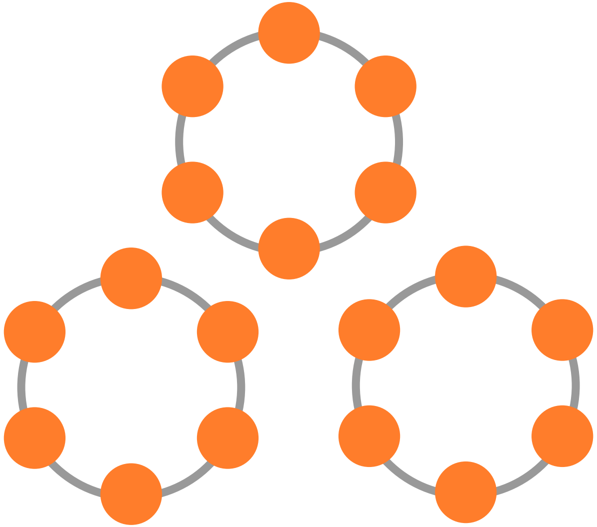
CAP Theorem

Any distributed data store can provide only **two of the three** guarantees

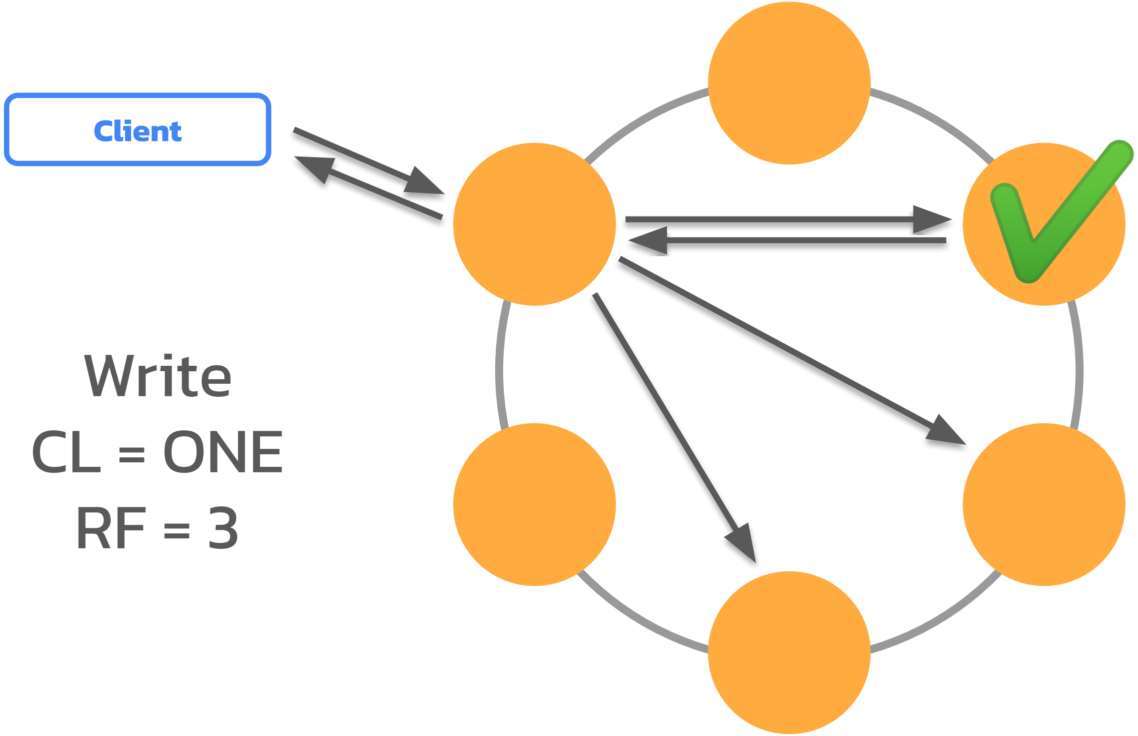


Tunable Consistency and Consistency Levels

- ANY
- ONE
- TWO
- THREE
- QUORUM
- LOCAL_ONE
- LOCAL_QUORUM
- EACH_QUORUM
- ALL



Consistency level ONE

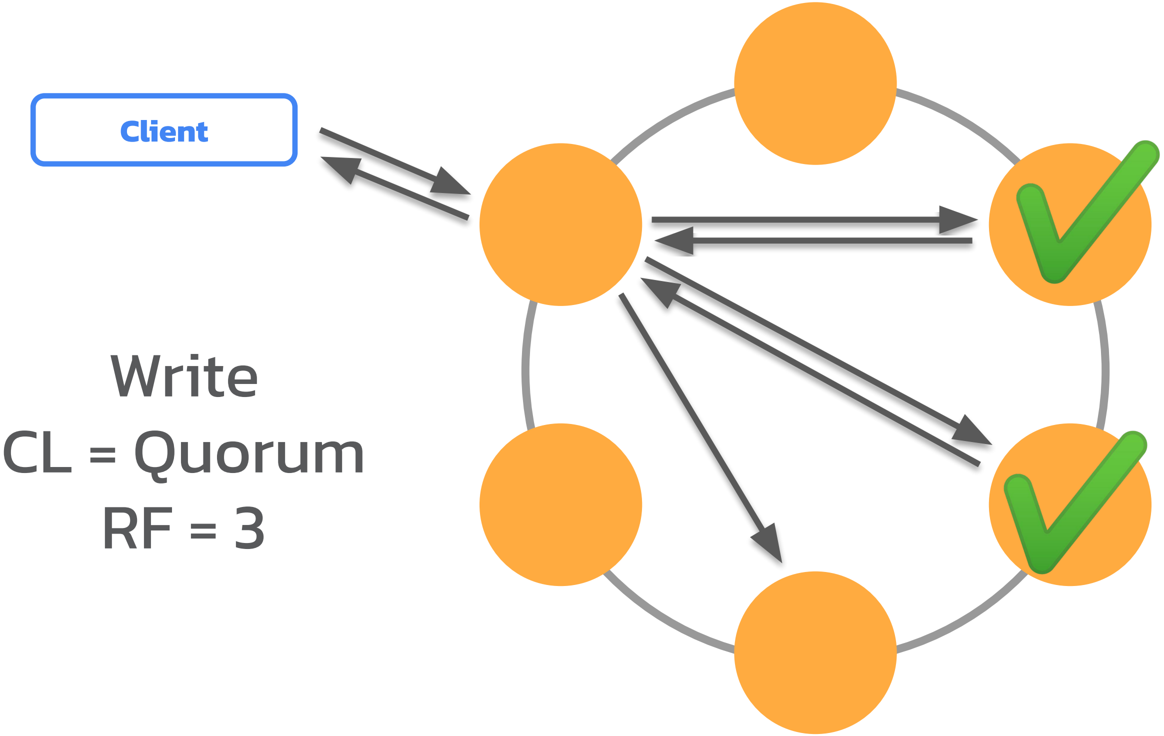


Write

CL = ONE

RF = 3

Consistency level QUORUM



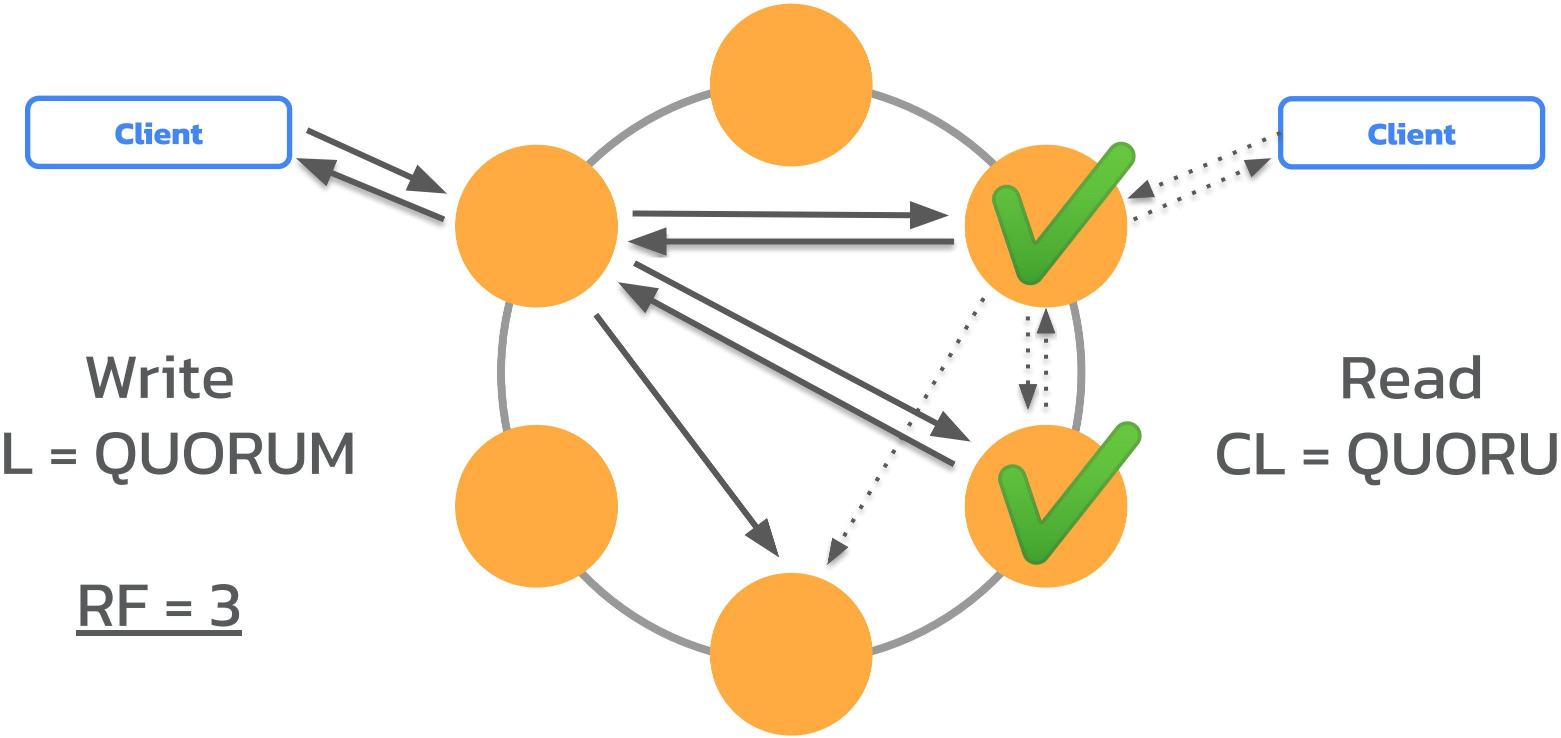
Write

CL = Quorum

RF = 3

Immediate Consistency

CL Write + CL Read > RF → Immediate Consistency



Write

CL = QUORUM

RF = 3

Read

CL = QUORUM

01

Getting started with
Apache Cassandra®

02

Hands-on lab 1:
Create a database

03

Data definition and
manipulation with CQL

04

Hands-on lab 2:
Create, populate and query

05

Application development
in Java and Python

06

Hands-on lab 3:
Create an app with C* driver

Lab 1

Create a Cassandra database instance in the cloud

Start here: `astra.datastax.com`

- ✅ Create database **workshops** and keyspace **sensor_data**
- ✅ Generate and save an application token

Follow a demo by the instructor to complete the steps

Astra = Cassandra As a Service++

Free to Use

Up to 80GB storage and/or 20 million operations monthly.

Serverless

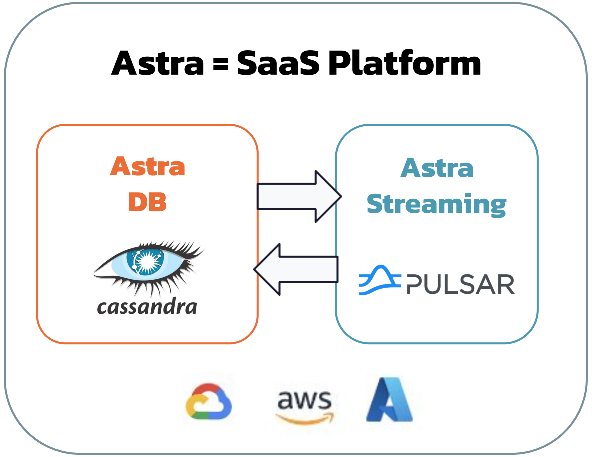
Lower your costs by running Cassandra clusters only when needed.

No Operations

Eliminate the overhead to install, operate, and scale Cassandra.

Data APIs

Work natively with Document (JSON), REST, GraphQL and gRPC APIs.



Global Scale

Put your data where you need it without compromising performance, availability or accessibility.

End-to-End Security

Secure connect with VPC peering and Private Link. Bring your own encryption key management. SAML SSO secure account access.

Zero Lock-in

Deploy on AWS, GCP or Azure and keep compatibility with open-source Cassandra.

Relational Indexes

Storage Attached Indexing (SAI) lets you query tables using any columns.

01

Getting started with
Apache Cassandra®

02

Hands-on lab 1:
Create a database

03

Data definition and
manipulation with CQL

04

Hands-on lab 2:
Create, populate and query

05

Application development
in Java and Python

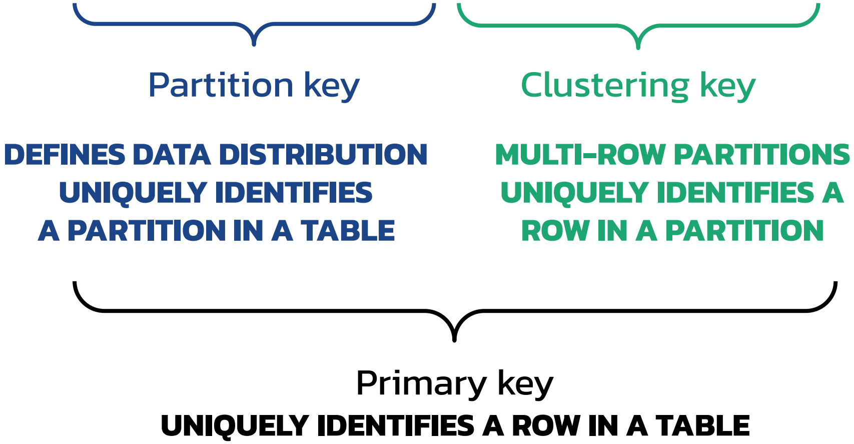
06

Hands-on lab 3:
Create an app with C* driver

Table definition and partitioning

Primary Key, Partition Key, Clustering Key

```
CREATE TABLE sensor_data.temperatures_by_sensor (
  sensor     TEXT,
  date       DATE,
  timestamp  TIMESTAMP,
  value      FLOAT,
  PRIMARY KEY (  (  sensor ,  date  ) ,     timestamp    )
) WITH CLUSTERING ORDER BY (timestamp DESC);
```



Partition key

DEFINES DATA DISTRIBUTION
UNIQUELY IDENTIFIES
A PARTITION IN A TABLE

Clustering key

MULTI-ROW PARTITIONS
UNIQUELY IDENTIFIES A
ROW IN A PARTITION

Primary key

UNIQUELY IDENTIFIES A ROW IN A TABLE

Table Structure ⇒ Valid Queries

```
PRIMARY KEY ((sensor, date), timestamp);
```

```
SELECT * FROM temperatures_by_sensor ...

    WHERE sensor = ?;

    WHERE sensor > ?;

    WHERE sensor = ? AND date > ?;

    WHERE sensor = ? AND date = ?;

    WHERE sensor = ? AND date = ? AND timestamp > ?;
```

Good Partition Rules

- **Store together what you retrieve together**
- Avoid big partitions
- Avoid hot partitions

Q: Show temperature evolution over time for sensor X On Nov 10, 2022

```
PRIMARY KEY ((sensor, timestamp));
```

✗

```
PRIMARY KEY ((sensor), timestamp);
```

✓

Good Partition Rules

- Store together what you retrieve together
- **Avoid big partitions**
- Avoid hot partitions

BUCKETING

```
PRIMARY KEY ((sensor), timestamp);
```

✗

```
PRIMARY KEY ((sensor, month_year), timestamp);
```

✓

- Up to 2 billion cells per partition
- Up to ~100k values in a partition
- Up to ~100MB in a Partition

Good Partition Rules

- ❖ Store together what you retrieve together
- ❖ Avoid big partitions
- ❖ **Avoid hot partitions**

```
PRIMARY KEY ((date), sensor, timestamp);
```

✗

```
PRIMARY KEY ((date, sensor), timestamp);
```

✓

```
PRIMARY KEY ((sensor, date), timestamp);
```

✓

Inserts, bulk loading, querying capabilities

Ingesting Data

- CQL statement `INSERT`
- CQL shell command `COPY FROM`
- Command-line utility `dsbulk`
- Apache Spark with Spark-Cassandra Connector

Querying Data

```
SELECT [DISTINCT] * |
       select_expression [AS column_name][ , ... ]
FROM   [keyspace_name.] table_name
[WHERE partition_key_predicate
  [AND clustering_key_predicate]]
[GROUP BY primary_key_column_name][ , ... ]
[ORDER BY clustering_key_column_name ASC|DESC][ , ... ]
[PER PARTITION LIMIT number]
[LIMIT number]
[ALLOW FILTERING]
```

01

Getting started with
Apache Cassandra®

02

Hands-on lab 1:
Create a database

03

Data definition and
manipulation with CQL

04

Hands-on lab 2:
Create, populate and query

05

Application development
in Java and Python

06

Hands-on lab 3:
Create an app with C* driver

Lab 2

Create, populate and query Cassandra tables

Start here: `tinyurl.com/cassandra-cql`

- ✅ Understand the data model and queries
- ✅ Create tables and run queries in Cassandra

Follow a demo by the instructor to complete the steps

01

Getting started with Apache Cassandra®

02

Hands-on lab 1: Create a database

03

Data definition and manipulation with CQL

04

Hands-on lab 2: Create, populate and query

05

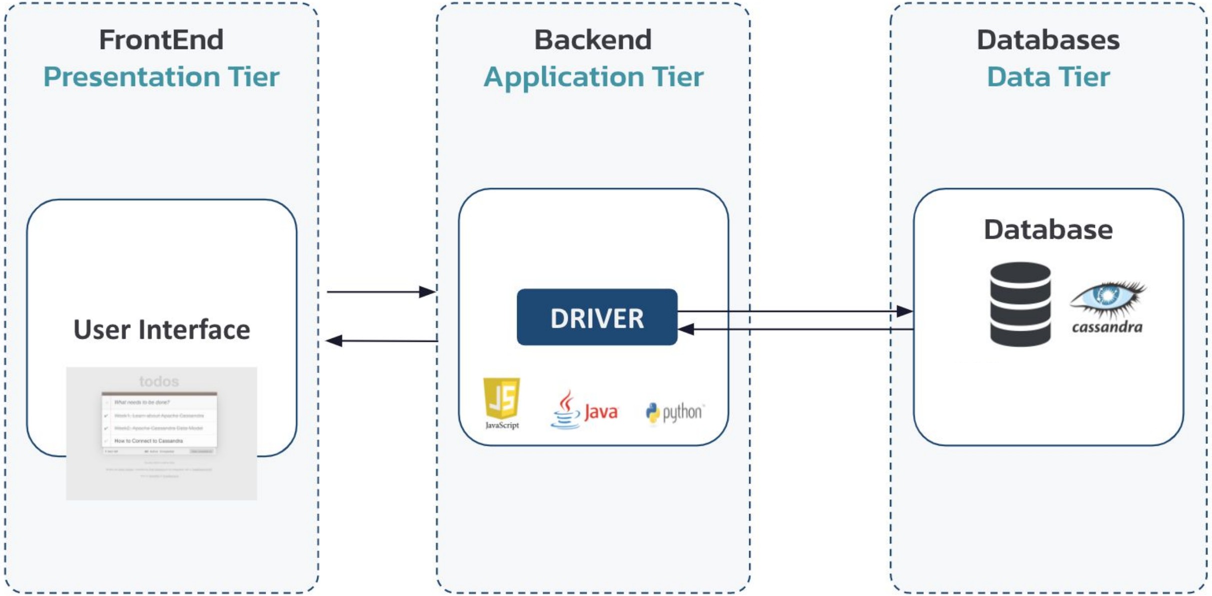
Application development in Java and Python

06

Hands-on lab 3: Create an app with C* driver

Cassandra Drivers

Application Development with Apache Cassandra®



FrontEnd
Presentation Tier

User Interface

todos

○ What needs to be done?

✓ ~~Week1: Learn about Apache Cassandra~~

✓ ~~Week2: Apache Cassandra Data Model~~

○ How to Connect to Cassandra

1 item left | All Active Completed | Clear completed

Backend
Application Tier

DRIVER

Databases
Data Tier

Database

Drivers

C C++ C# ODBC node js python Java

Connectivity

- Token & Datacenter Aware
- Load Balancing Policies
- Retry Policies
- Reconnection Policies
- Connection Pooling
- Health Checks
- Authentication | Authorization
- SSL

Query

- CQL Support
- Schema Management
- Sync/Async/Reactive API
- Query Builder
- Compression
- Paging

Parsing Results

- Lazy Load
- Object Mapper
- Spring Support
- Paging

Installing the Drivers

```
<dependency>
    <groupId>com.datastax.oss</groupId>
    <artifactId>java-driver-core</artifactId>
    <version>4.13.1</version>
</dependency>
```

Maven

Java

```
npm install cassandra-driver
```

4.6.3

```
{
  "dependencies": {
      "cassandra-driver": "^4.6.3"
  }
}
```

JavaScript

python Package Index

```
pip install cassandra-driver==3.25.0
```

python

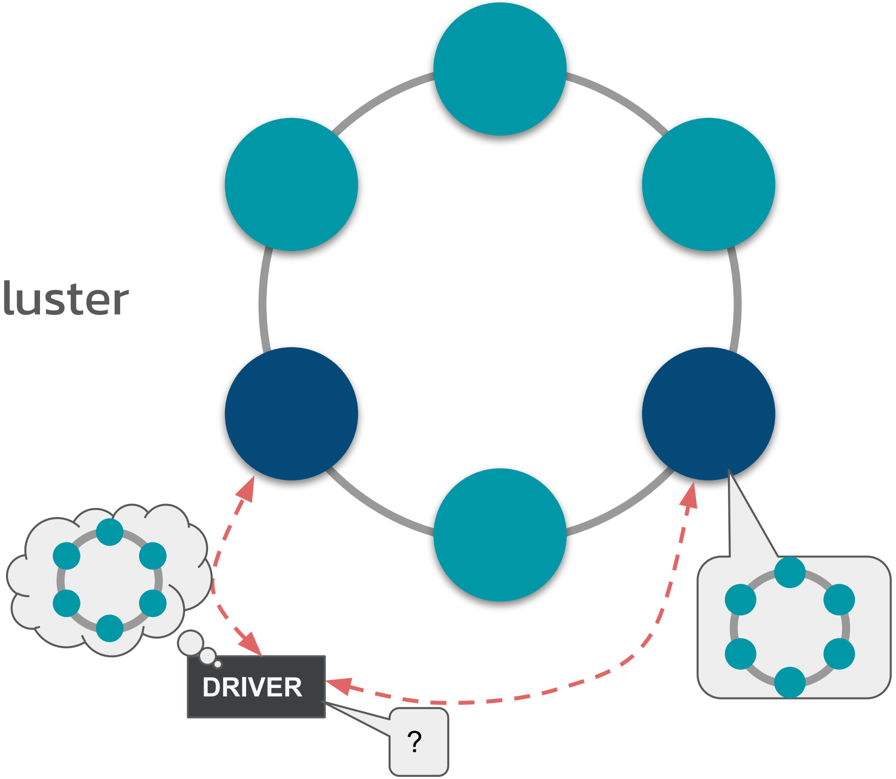
nuget v3.15.0

```
Install-Package CassandraCSharpDriver -Version 3.15.0
```

Contact Points (Cassandra)

- One contact point *would be enough*

  … unless that node is down
- ~3 nodes per DC for resilience
- From there, drivers discover whole cluster
- Local Datacenter



Create "Session" Client (with contact points)

```java
CqlSession cqlSession = CqlSession.builder()
 .addContactPoint(new InetSocketAddress("127.0.0.1", 9042))
 .withKeyspace("sensor_data")
 .withLocalDatacenter("dc1")
 .withAuthCredentials("U","P")
 .build();
```

```javascript
const client = new cassandra.Client({
  contactPoints: ['127.0.0.1'],
  localDataCenter: 'dc1',
  keyspace: 'sensor_data',
  credentials: { username: 'U', password: 'P' }
});
await client.connect();
```

```python
auth_provider = PlainTextAuthProvider(
   username='U', password='P')
cluster = Cluster(['127.0.0.1'],
  auth_provider=auth_provider, protocol_version=5)
session = cluster.connect('sensor_data')
```

```csharp
Cluster cluster = Cluster.Builder()
.AddContactPoint("127.0.0.1")
.WithCredentials("U", "P")
.Build();
session = cluster.Connect("sensor_data");
```

Create "Session" Client (with Astra DB)

```java
CqlSession cqlSession = CqlSession.builder()
 .withCloudSecureConnectBundle(Paths.get("secure.zip"))
 .withAuthCredentials("U","P")
 .withKeyspace("sensor_data")
 .build();
```

```javascript
const client = new cassandra.Client({
   cloud: { secureConnectBundle: 'secure.zip' },
   credentials: { username: 'u', password: 'p' },
   keyspace: 'sensor_data'
});
await client.connect();
```

```python
auth_provider = PlainTextAuthProvider(
    username='U', password='P')
cluster = Cluster(
    cloud ={'secure_connect_bundle': 'secure.zip'},
   auth_provider=auth_provider, protocol_version=4)
session= cluster.connect('sensor_data')
```

```csharp
var cluster = Cluster.Builder()
 .WithCloudSecureConnectionBundle("secure.zip")
 .WithCredentials("u", "p")
 .Build();
var session = cluster.Connect("sensor_data");
```

There Should Only Be One Session !

- Stateful object handling communications with each node
- Should be unique in the Application (*Singleton*)
- Should be closed at application shutdown (*shutdown hook*) in order to free opened TCP sockets (*stateful*)

```
Java:       cqlSession.close();
Python:     session.shutdown();
Node:       client.shutdown();
CSharp:     IDisposable
```

Executing CQL Queries

python™

```
session.execute(
    "SELECT * FROM sensors_by_network WHERE network = %s;",
    (network,)
)
```

Java

```
cqlSession.execute(
    "SELECT * FROM sensors_by_network WHERE network = '" + network + "'"
);
```

Prepared CQL Statements

python™

```
q3_statement = session.prepare(
    "SELECT * FROM sensors_by_network WHERE network = ?;"
)
rows = session.execute(q3_statement, (network,) )
```

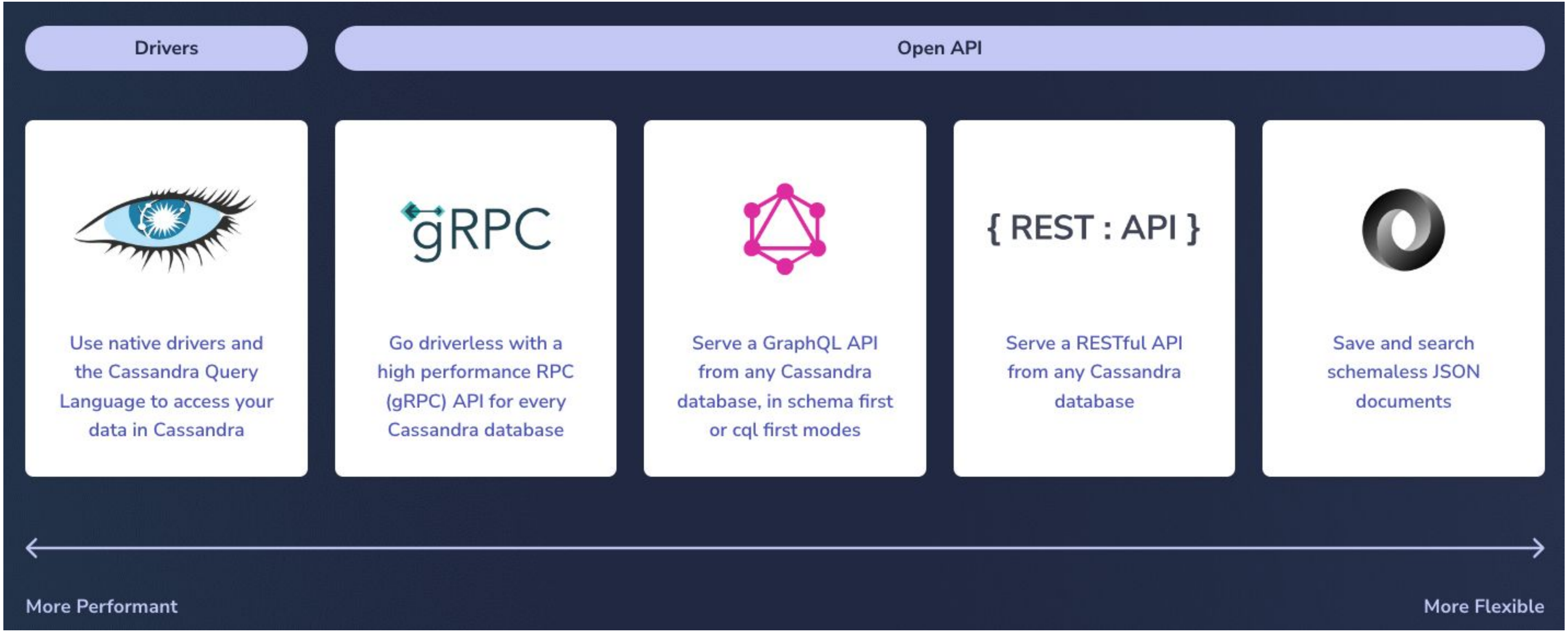
Java

```
PreparedStatement q3Prepared = session.prepare(
    "SELECT * FROM sensors_by_network WHERE network = ?");
BoundStatement q3Bound = q3Prepared.bind(network);
ResultSet rs = session.execute(q3Bound);
```

Data APIs

Open Source Data API Gateway

Stargate.io



Drivers

Open API

Use native drivers and the Cassandra Query Language to access your data in Cassandra

Go driverless with a high performance RPC (gRPC) API for every Cassandra database

Serve a GraphQL API from any Cassandra database, in schema first or cql first modes

Serve a RESTful API from any Cassandra database

Save and search schemaless JSON documents

More Performant

More Flexible

01

Getting started with
Apache Cassandra®

02

Hands-on lab 1:
Create a database

03

Data definition and
manipulation with CQL

04

Hands-on lab 2:
Create, populate and query

05

Application development
in Java and Python

06

Hands-on lab 3:
Create an app with C* driver

Lab 3

Create a Java or Python app
to query a Cassandra database

Start here: `tinyurl.com/cassandra-dev`

✅ Explore and deploy a Java app (+ Spring Boot)

✅ Explore and deploy a Python app (+ Fast API)

Follow the steps at the link

Stay in touch!

Discord:	dtsx.io/discord
Academy:	academy.datastax.com
Workshops:	datastax.com/workshops
YouTube:	@DataStaxDevelopers

Get Cassandra help

stackoverflow (*):

stackoverflow.com/questions/tagged/cassandra

DBA Stack Exchange(*):

dba.stackexchange.com/questions/tagged/cassandra

Discord:

dtsx.io/discord

(*) for best results, follow the "cassandra" tag

Sponsored by

DataStax

Thank You