Sponsored by

DataStax

intel

Cassandra Day

Discover the real power of NoSQL

Berlin – London – Amsterdam – Santa Clara – Seattle – Houston – Hanoi – Jakarta – Singapore

Agenda

Santa Clara

12:00 - 1:00 pm	*Check-in for Hands-on Workshop + Lunch*	
1:00 - 3:00 pm	**Hands-on Workshop: Building data-driven applications with NoSQL and Apache Cassandra®**	
3:00 - 3:30 pm	*Meetup check-in + Snacks and drinks*	
3:30 - 4:00 pm	**Intel's contributions to a faster Cassandra**	**| Smita Kamath, Shylaja Kokoori**
4:00 - 4:30 pm	**Apache Cassandra® in 2022: What to Expect from 4.1 and Beyond**	**| Scott Andreas**
4:30 - 5:00 pm	**Cassandra Performance Tuning, Tricks, and Tools**	**| Jon Haddad**
5:00 - 5:30 pm	**ACID transactions in Apache Cassandra®**	**| Patrick McFadin**
5:30 - 6:00 pm	*Food & Networking*	
6:00 - 7:00 pm	*Travel to AMC Mercado 20 - 3111 Mission College Blvd, Santa Clara*	
7:00 - 10:00 pm	**Black Panther: Wakanda Forever**	

WiFi

Wifi Name: Hyatt_Meeting
Password: stax22

Discord: stay connected at dtsx.io/discord

Housekeeping

Health & Safety Measures

Snacks, Food & Drinks

Network

Share Your Experience

Feedback Form

Freebees & Rewards

Freebies & Rewards

Everyone attending Cassandra Day will “take home”:

- ★ $300 free credits to use on Astra DB (i.e. Cassandra-as-a-Service)
- ★ *Apache Cassandra™ Certification Exam* Voucher (Regular Price $145)
- ★ Apache Cassandra™ T-Shirt (Priceless)

The Event Sponsors

Apache Cassandra®
Hands-On Workshop

November 10, 2022

Sponsored by DataStax

Aleksandr Volochnev

Developer Advocate Lead

dtsx.io/aleks

@aleks-volochnev

@HadesArchitect

Stefano Lottini

Developer Advocate

dtsx.io/stefano

@stefano-lottini

@hemidactylus

@rsprrs

Your instructors

@ArtemChebotko

- Data professional, computer scientist
- Data modeling, data quality,
  data warehousing, data analytics
- Author of the Cassandra Data Modeling Methodology
- Google Cloud Certified Data Engineer

Your Instructor: Artem Chebotko

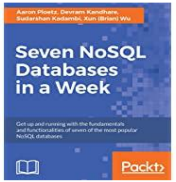
DataStax

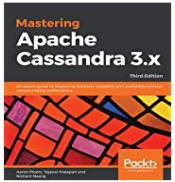
@aaronploetz

@aarOnp

@aploetz

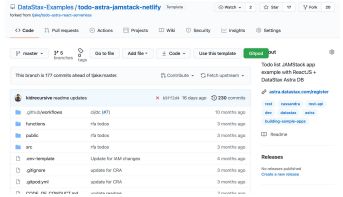
- Former SWE/DevOps/DB Lead @ GRAINGER & TARGET
- Host – Apache Cassandra Corner podcast
- Worked as an author on:
  - Mastering Apache Cassandra 3.x
  - Seven NoSQL Databases in a Week





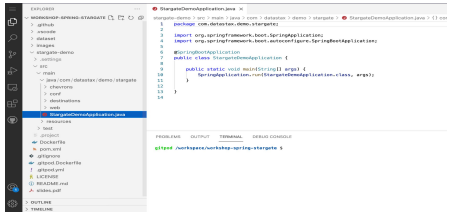
Your Instructor: Aaron Ploetz

Source code + exercises + slides



GitHub

IDE



Gitpod

Nothing to install !

Database + Api + Streaming

DataStax Astra DB

Hands-On Housekeeping

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

Agenda

The most scalable NoSQL database

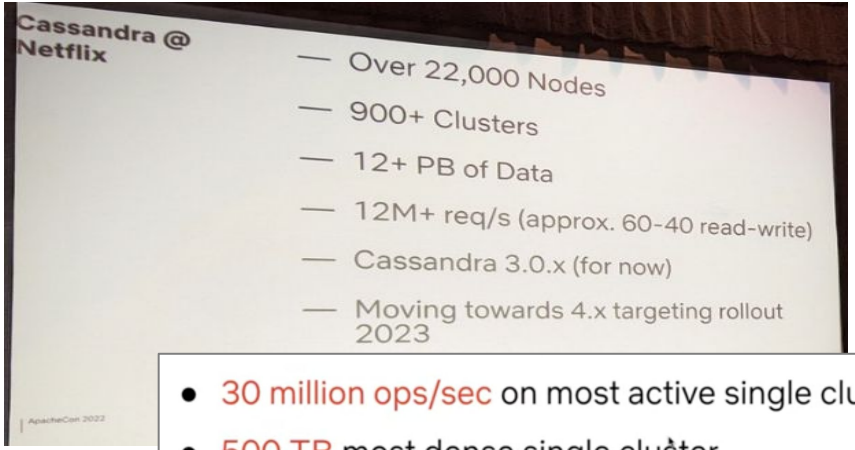
Today's Scope:

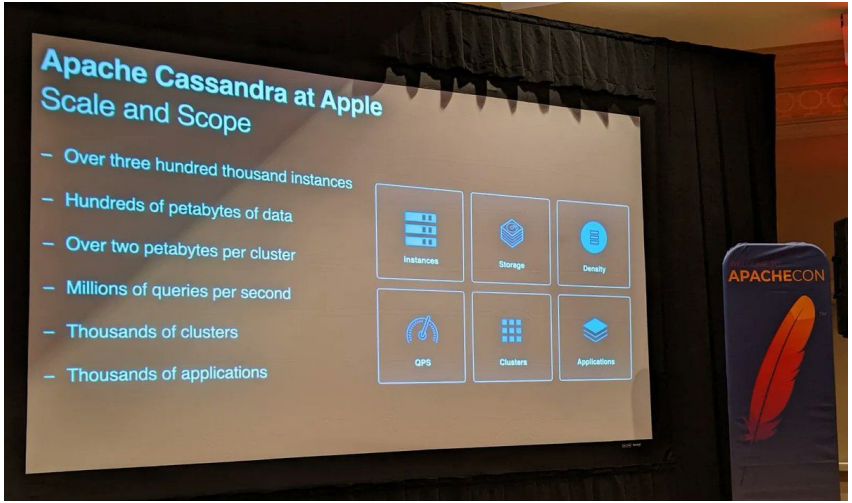
Show you how to build simple applications with Cassandra and AstraDB.

Apache Cassandra

NoSQL Distributed Decentralised Database Management System

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





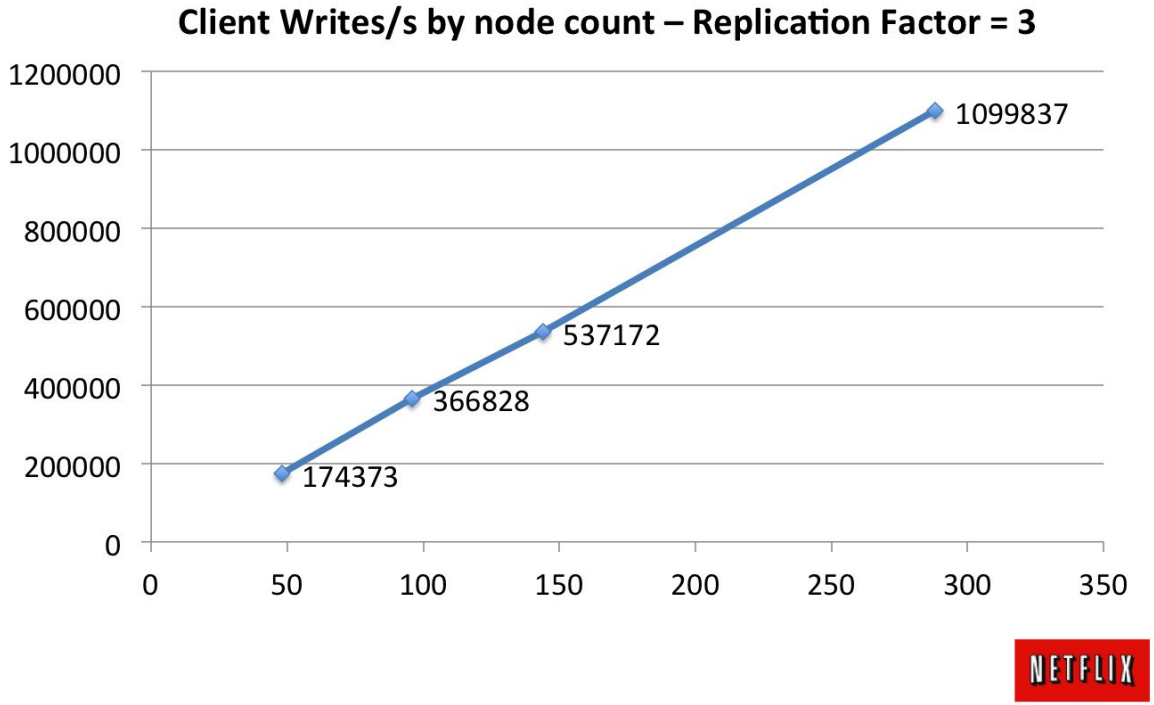
- 30 million ops/sec on most active single cluster
- 500 TB most dense single cluster
- 9216 CPUs in biggest cluster

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

And many others...

Cassandra Biggest Users (and Developers)

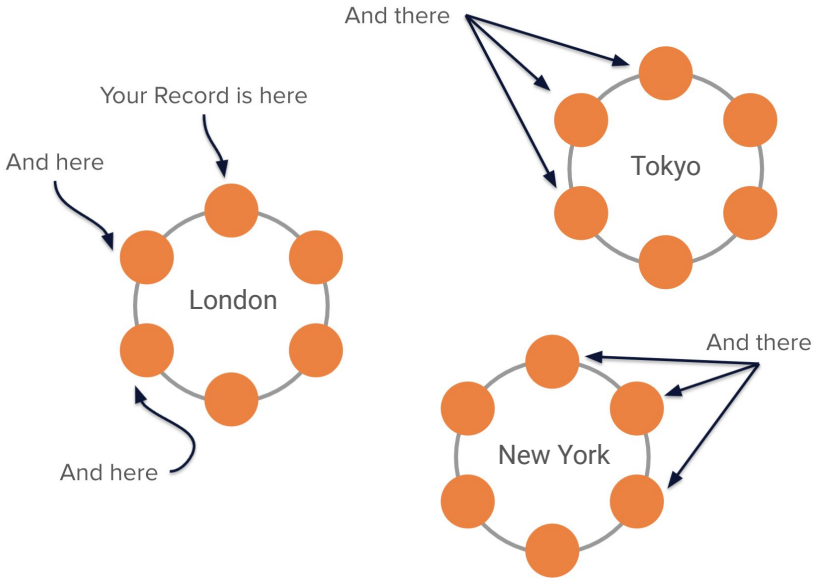


NETFLIX

Linear Scalability

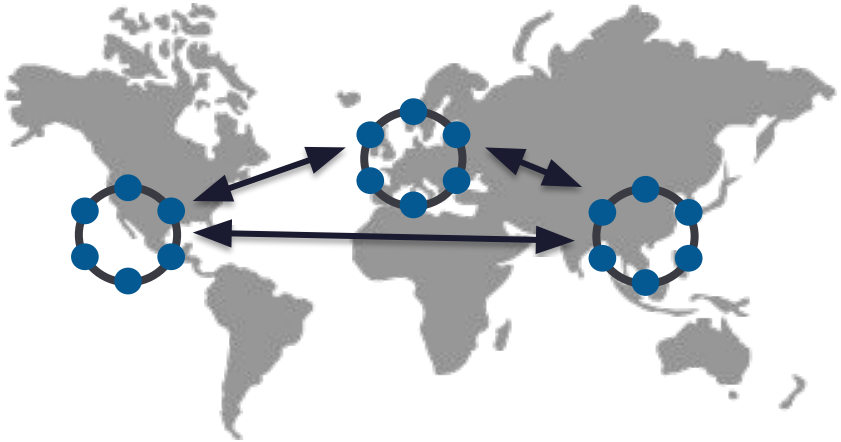
Replication, Decentralisation, and Topology-Aware Placement Strategy take care of possible downtimes:

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



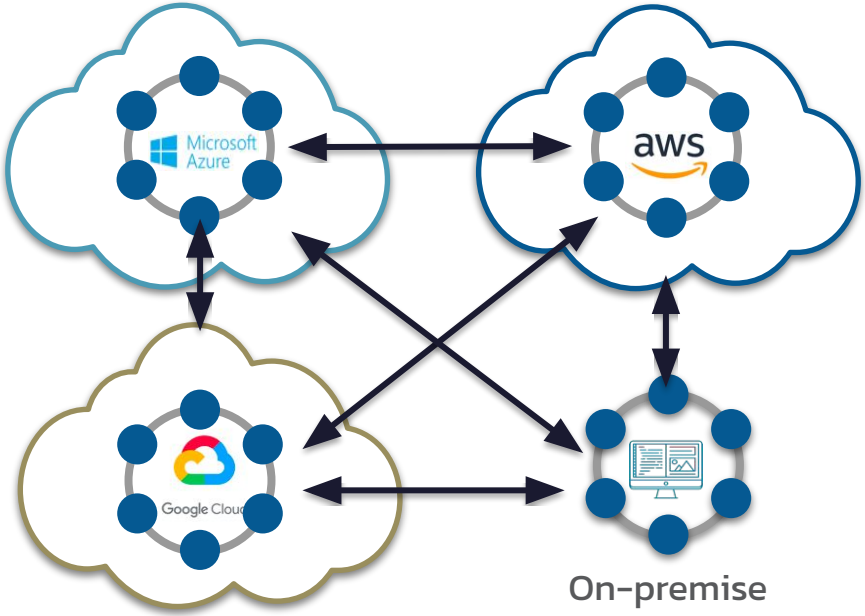
Cassandra's trademark is multi-datacenter deployments, granting you an exceptional capability for disaster tolerance while keeping your data close to your clients - worldwide.

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



Geographical Distribution

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

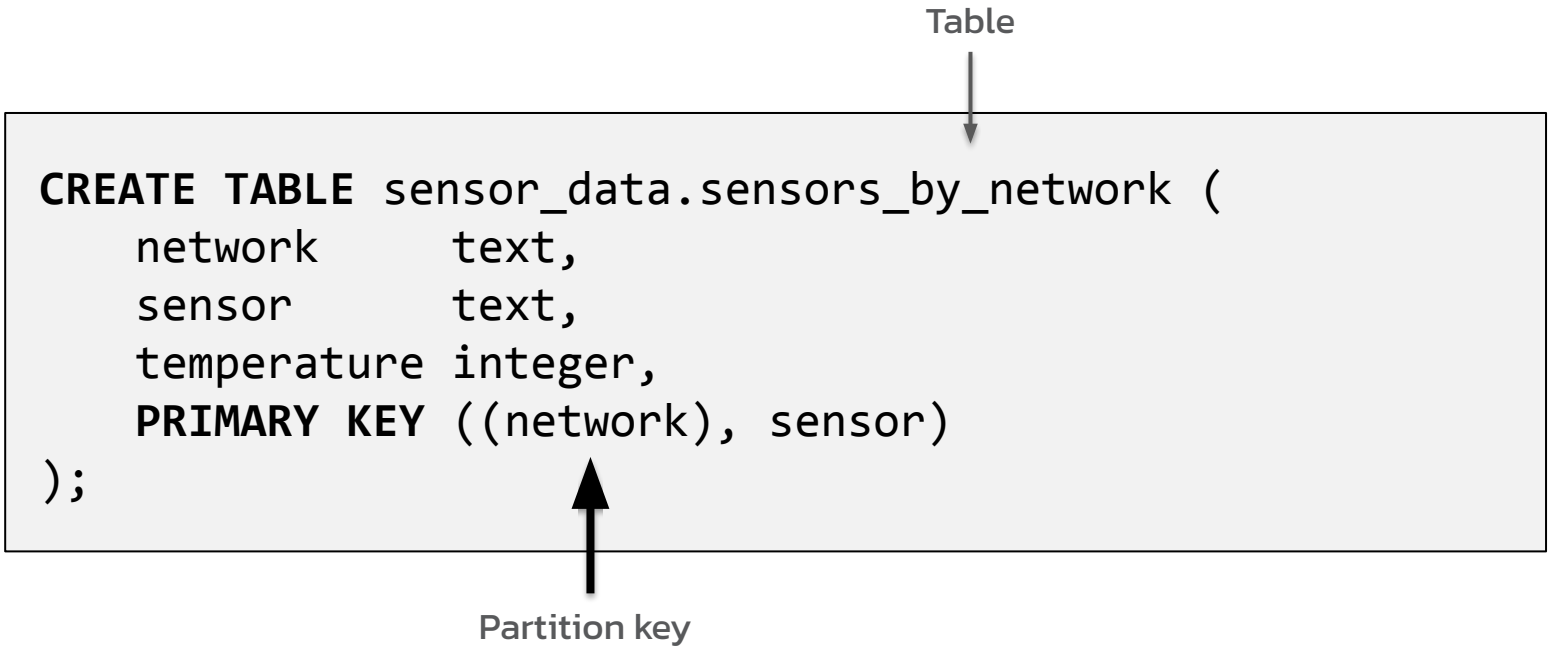


Platform Agnostic

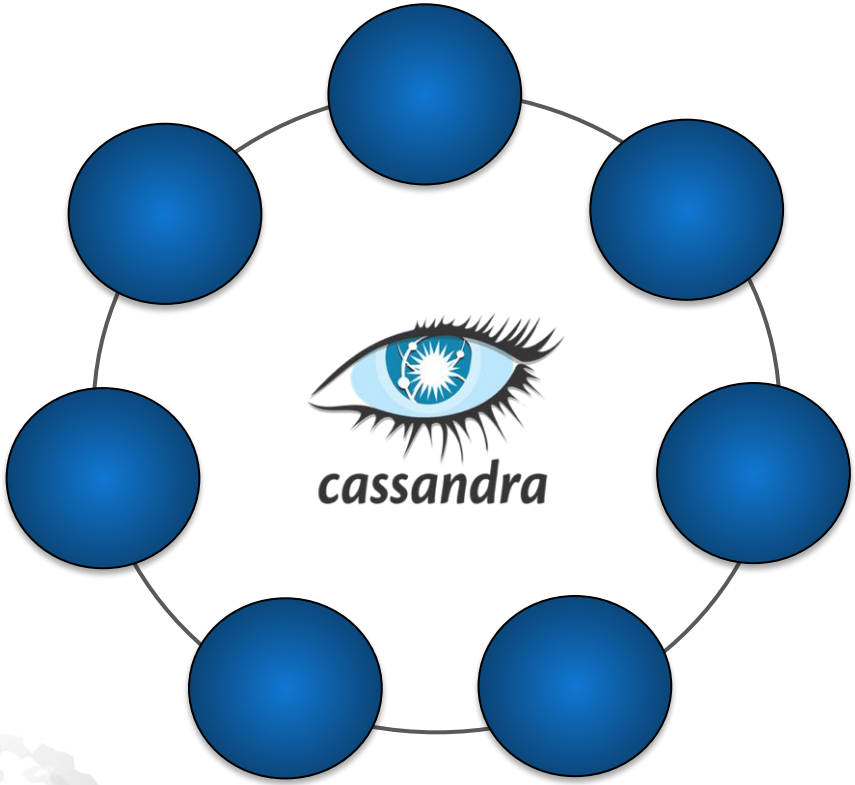
Cassandra doesn't belong to any of commercial vendors but controlled by a non-profit Open Source **Apache Software Foundation**, already familiar to you by *Hadoop*, *Spark*, *Kafka*, *Zookeeper, Maven* and many other projects.

Vendor Independent

Data distribution, replication and consistency



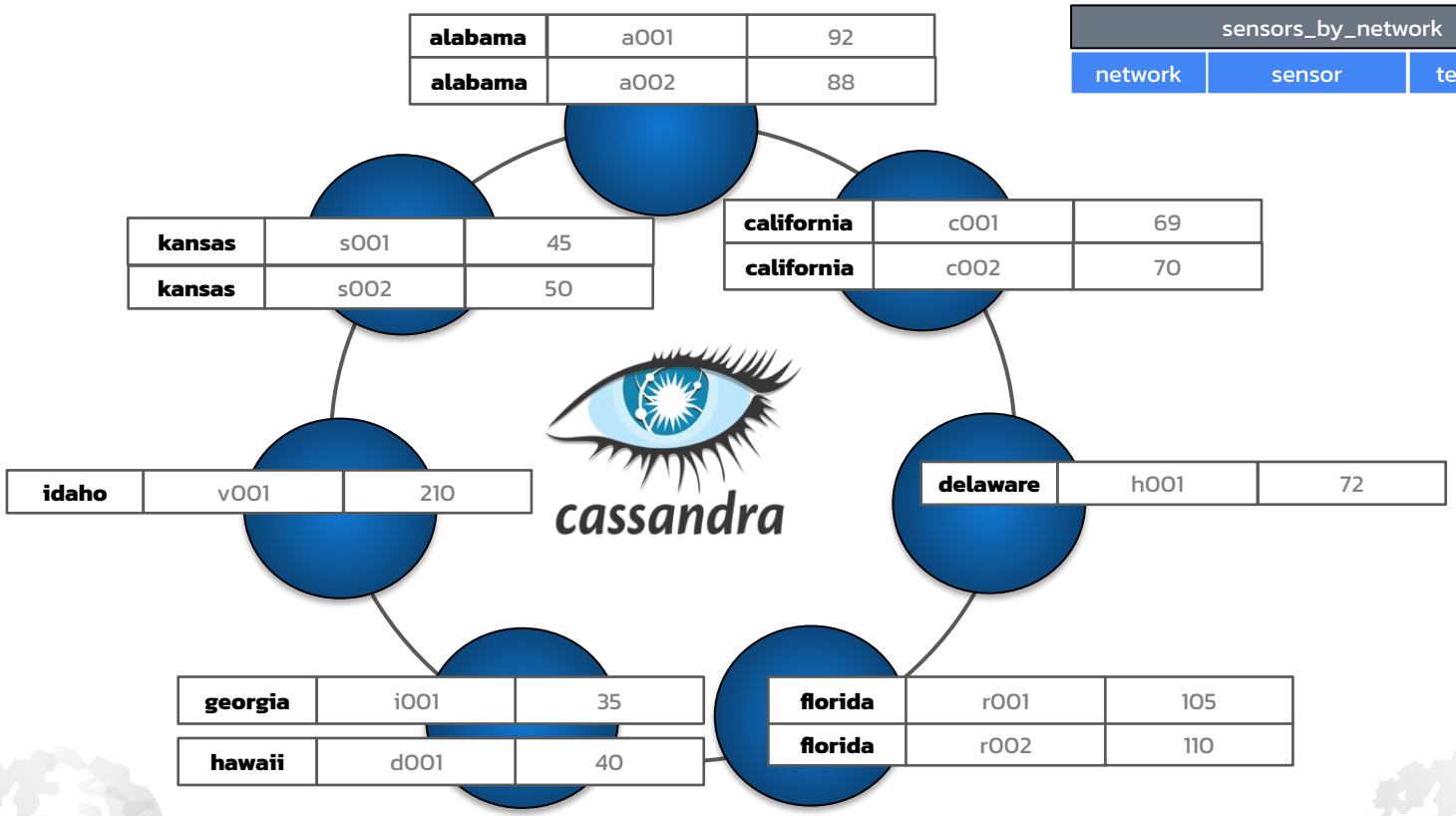
Partition key definition in CQL



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

Partition key and partitions



Data distribution

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

Partition Keys

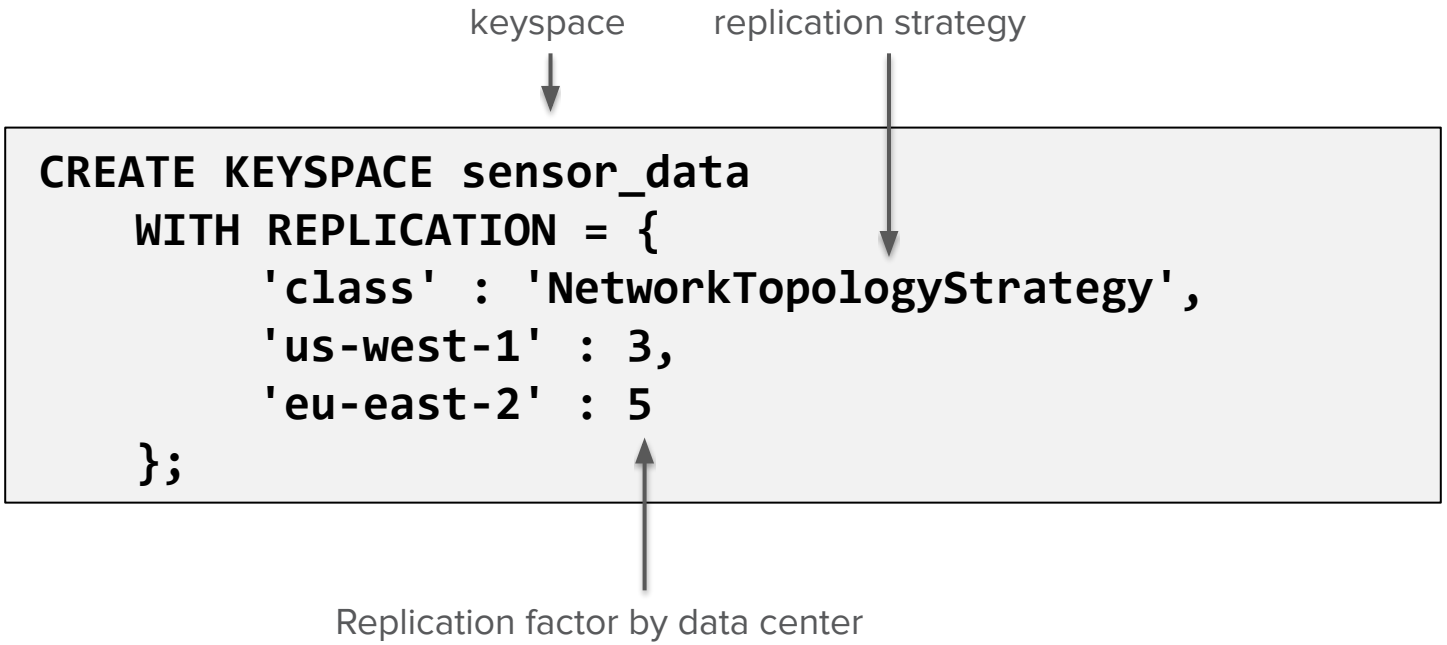
Partitioner

Murmur3 Hashing

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

Tokens

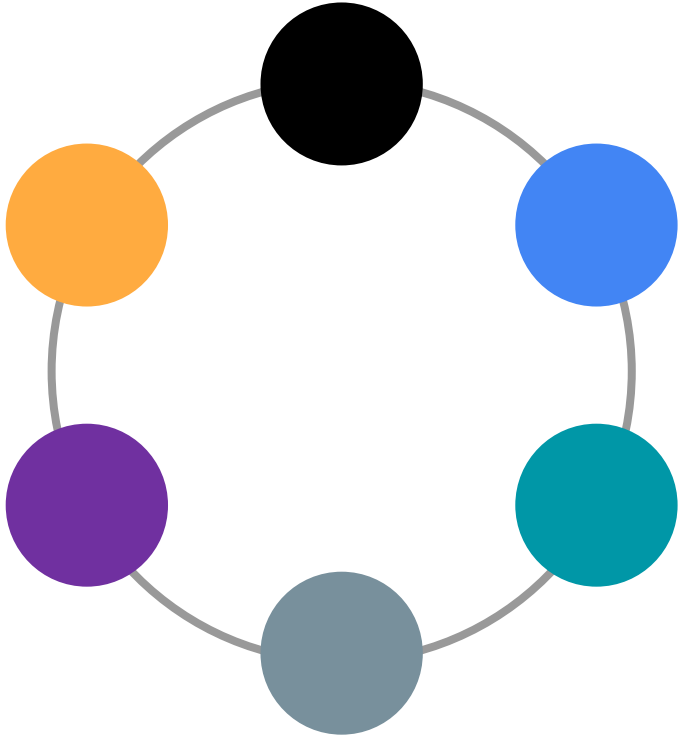
Cassandra Nodes



Replication is defined per keyspace

RF = ?

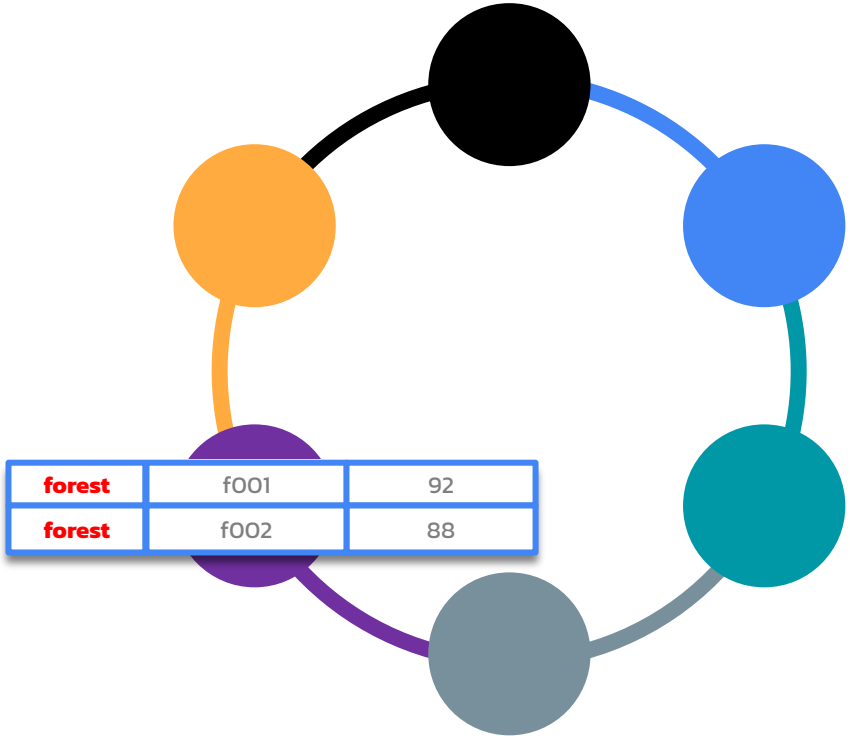
Replication Factor means the number of nodes used to store each partition



Replication Factor

RF = 1

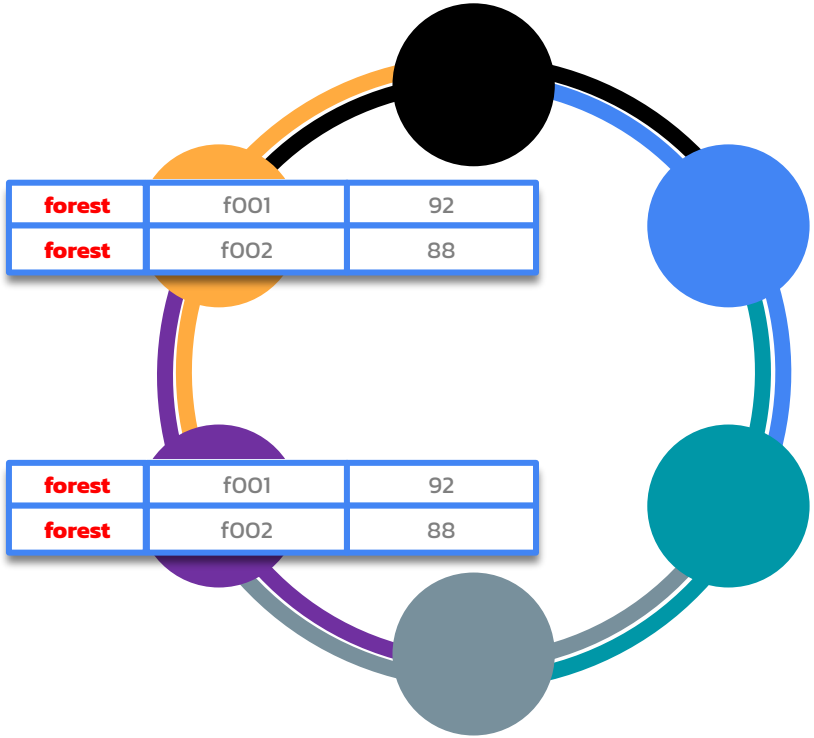
Replication Factor 1 means that every partition is stored on 1 node



RF = 1

RF = 2

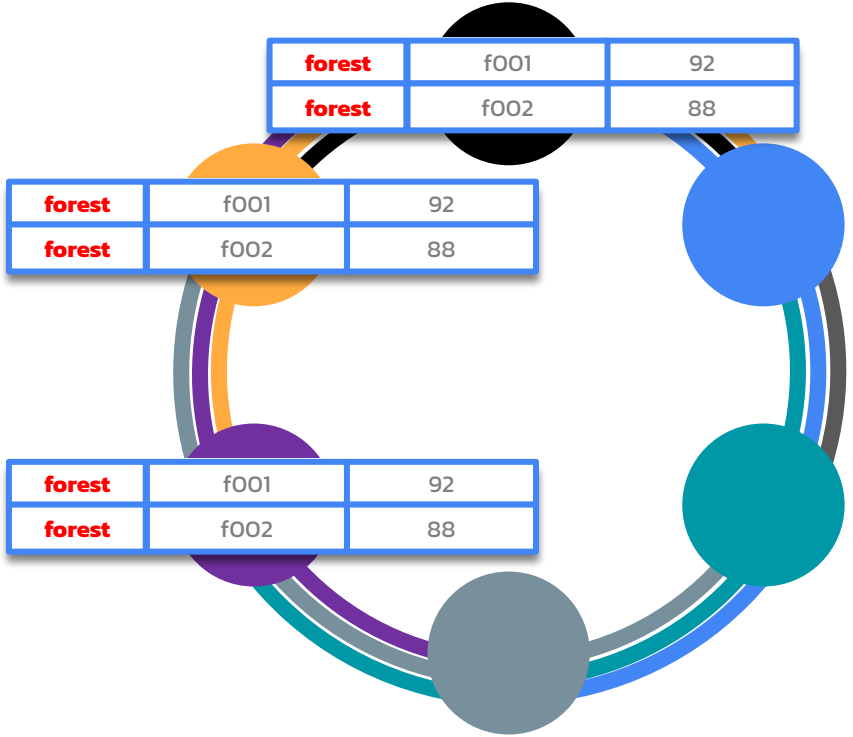
Replication Factor 2 means that every partition is stored on 2 nodes



RF = 2

RF = 3

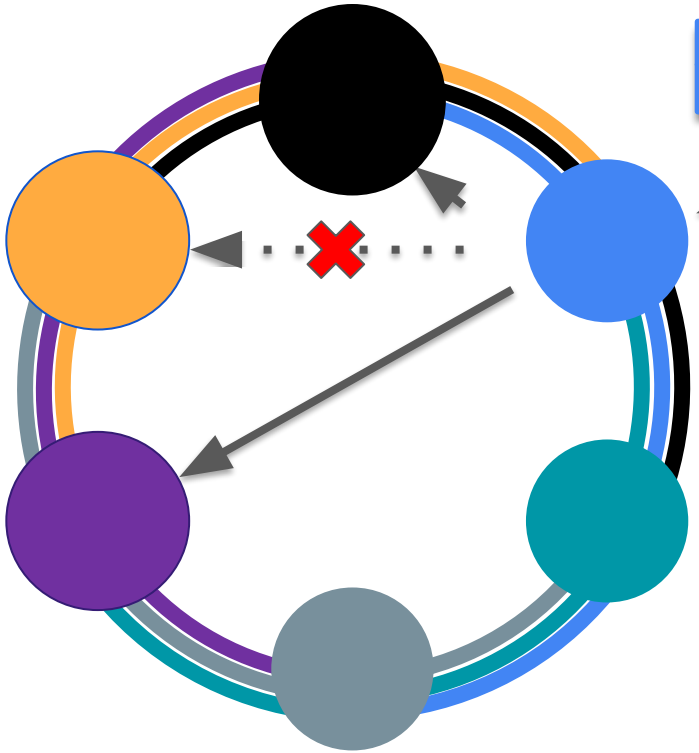
Replication Factor 3 means that every partition is stored on 3 nodes



RF = 3

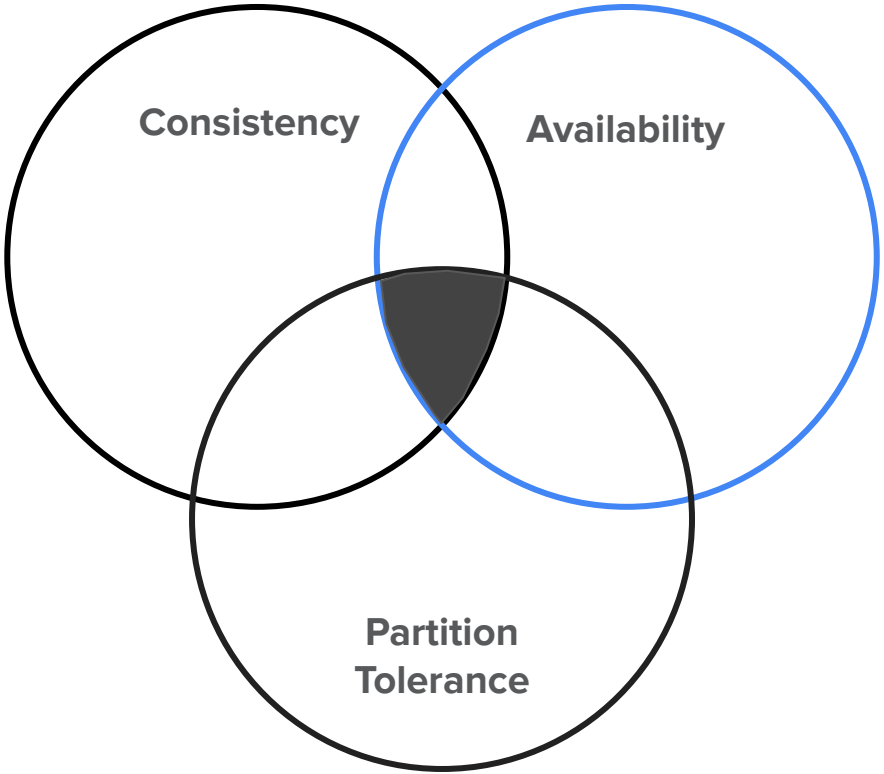
But what if ...

RF = 3

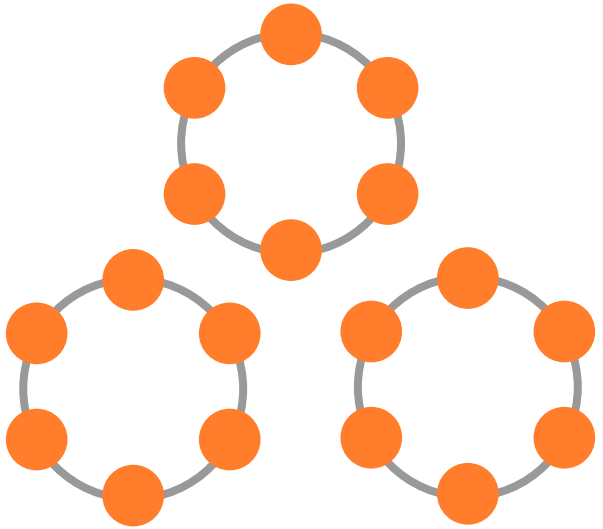


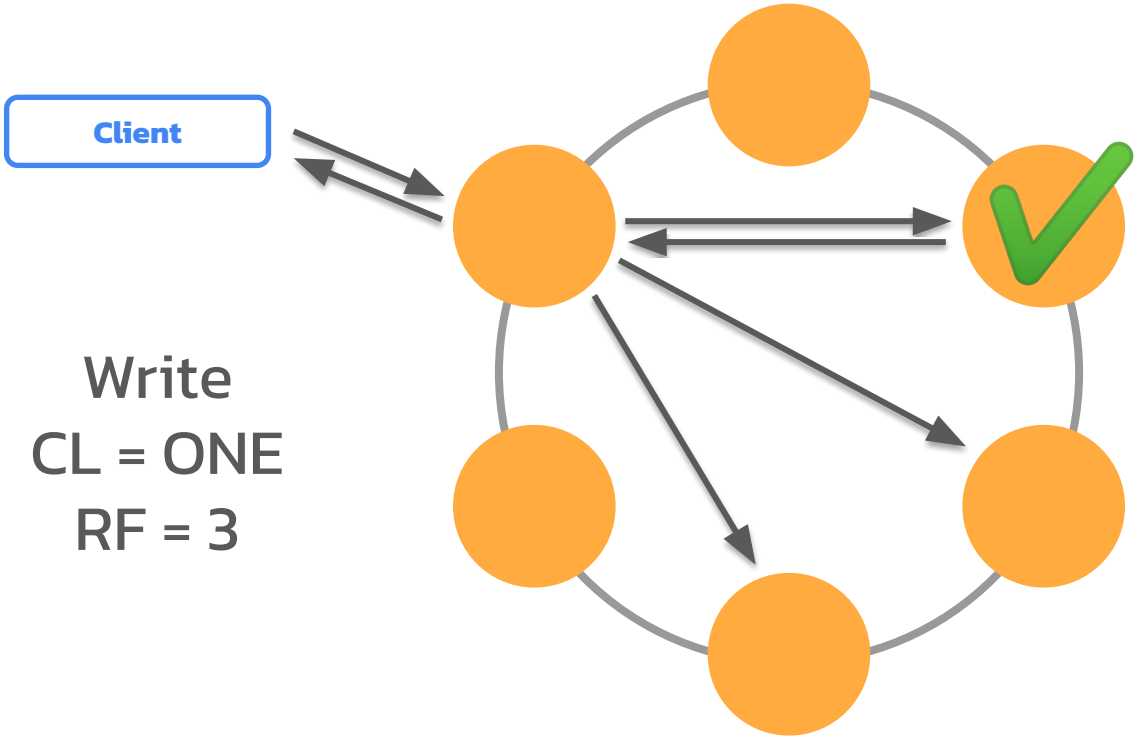
forest	f001	92
forest	f002	88

In the distributed environment **in case of emergency** you can have only two guaranteed qualities out of three :(



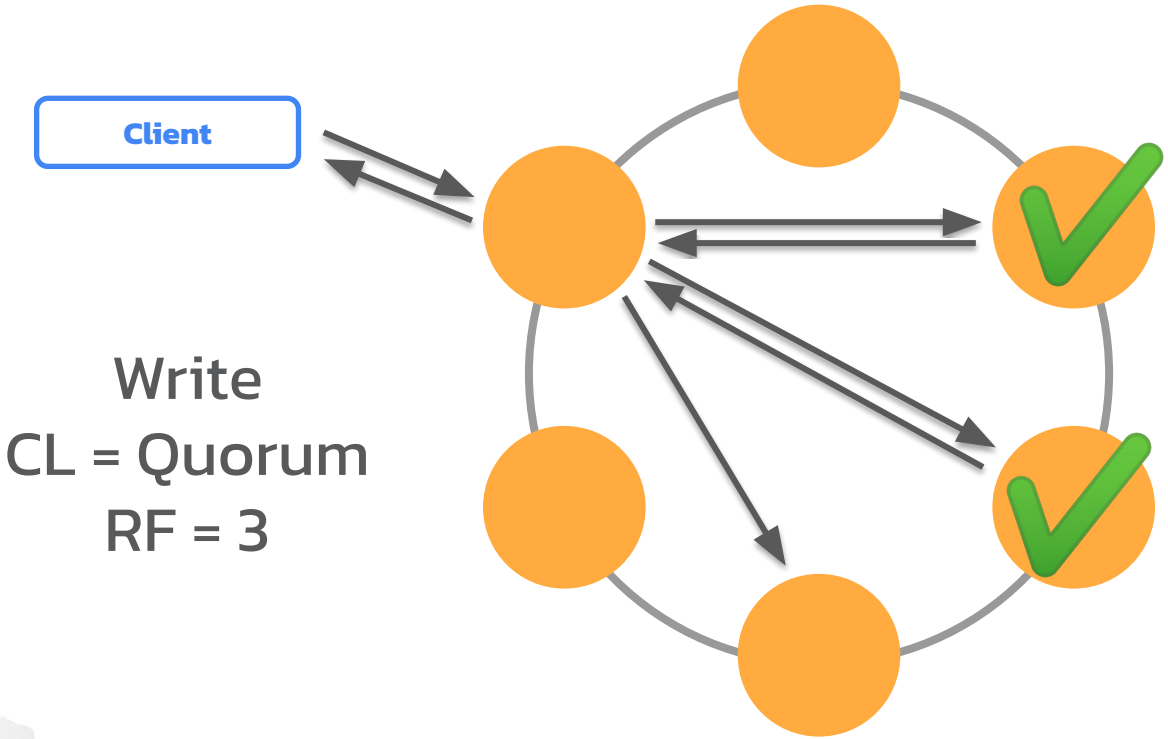
- ANY
- ONE
- TWO
- THREE
- QUORUM
- LOCAL_ONE
- LOCAL_QUORUM
- EACH_QUORUM
- ALL





Write
CL = ONE
RF = 3

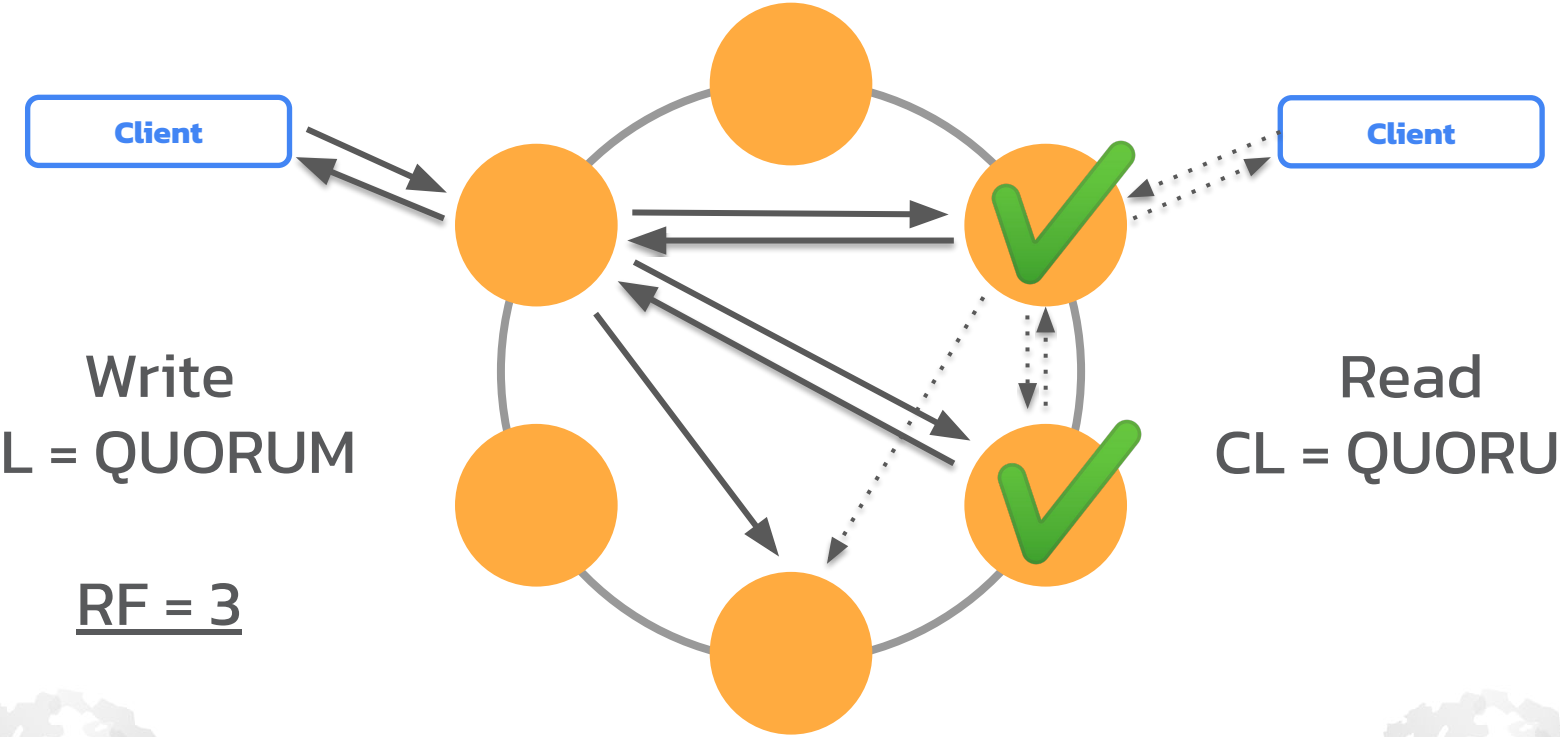
Consistency Level ONE



Write
CL = Quorum
RF = 3

Consistency Level QUORUM

CL Write + CL Read > RF → Immediate Consistency



Immediate Consistency

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

astra.datastax.com

✅ Create database workshops and keyspace sensor_data

✅ Generate and save an application token

https://github.com/datastaxdevs/workshop-cassandra-application-development

Free to Use

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

Serverless

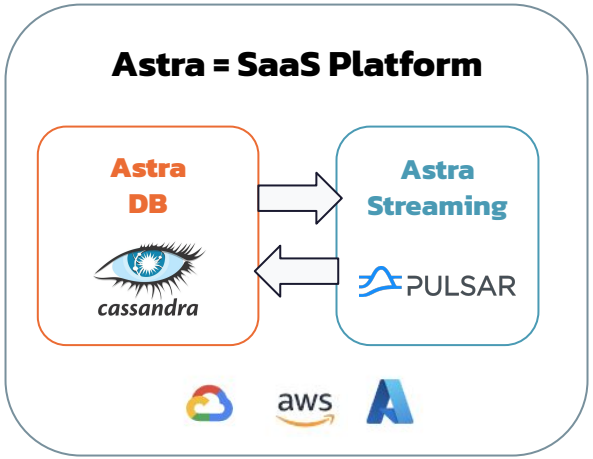
Lower your costs by running Cassandra clusters only when needed.

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.

Astra = Cassandra As a Service++

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);
```

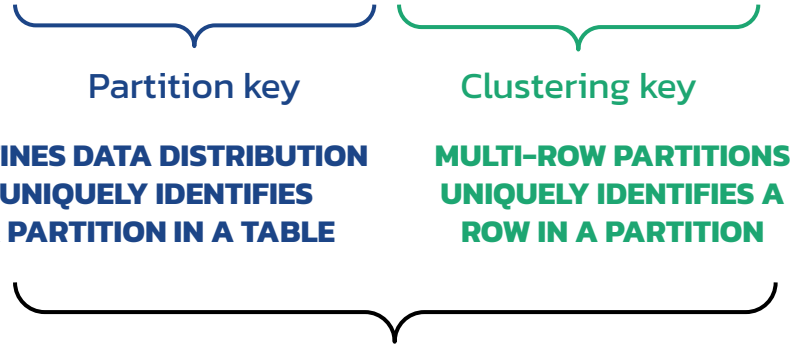


Table Structure ⇒ Valid Queries

```
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

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

https://github.com/datastaxdevs/workshop-cassandra-application-development

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

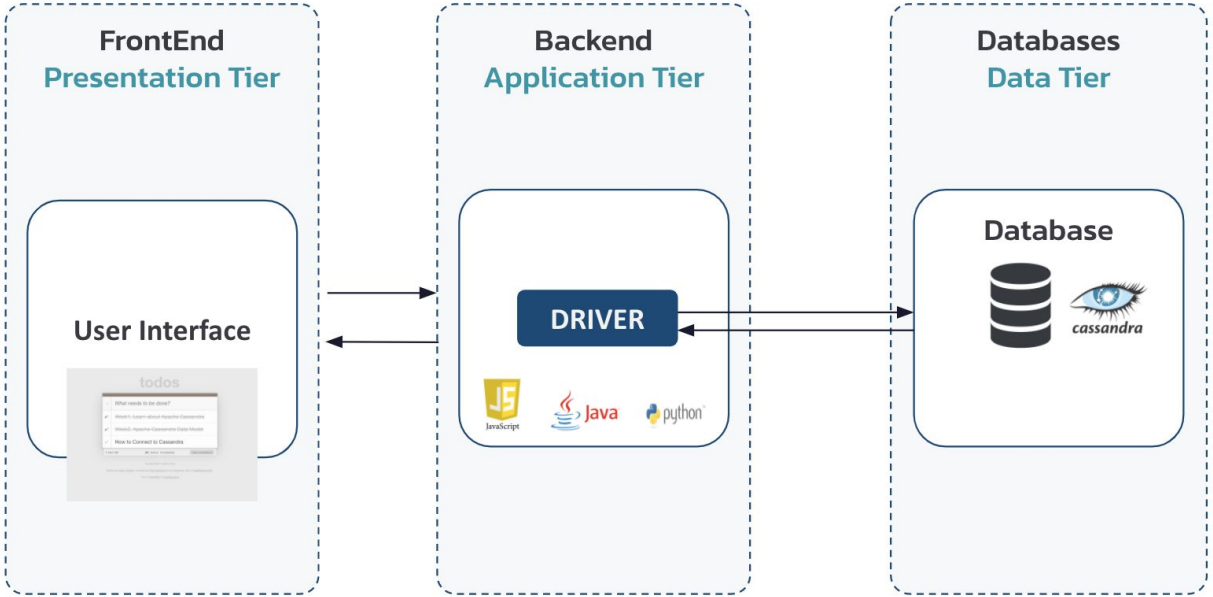
Application development in Java and Python

06

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

Cassandra Drivers

Application Development with Apache Cassandra®



Drivers






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

Maven

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

Java

```
npm install cassandra-driver
```

4.6.3

npm

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

JavaScript

python Package Index

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

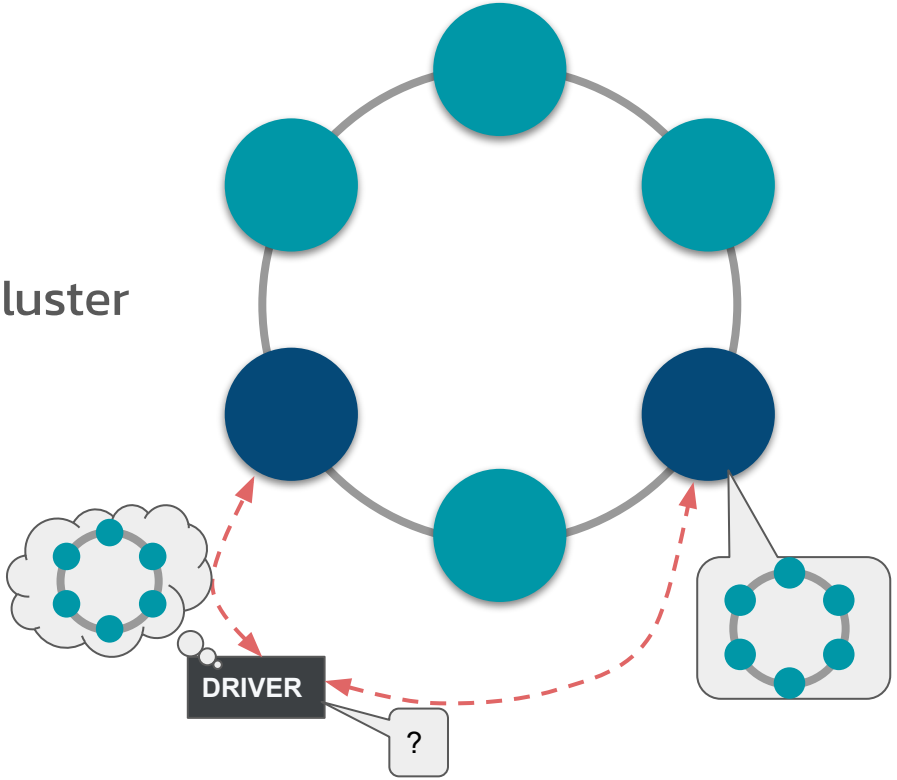
python

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();
```


```node
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')
```


```dotnet
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();
```


```node
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

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

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

Asynchronous CQL Queries


```python
# build a list of futures
futures = []
query = "SELECT * FROM users WHERE user_id=%s"
for user_id in ids_to_fetch:
    futures.append(session.execute_async(query, [user_id])

# wait for them to complete and use the results
for future in futures:
    rows = future.result()
     print rows[0].name
```

Asynchronous CQL Queries

Java

```
CompletionStage<CqlSession> sessionStage = CqlSession.builder().buildAsync();

// Chain one async operation after another:

CompletionStage<AsyncResultSet> responseStage = sessionStage.thenCompose(

        session -> session.executeAsync("SELECT release_version FROM system.local"));

// Apply a synchronous computation:

CompletionStage<String> resultStage =

    responseStage.thenApply(resultSet -> resultSet.one().getString("release_version"));

// Perform an action once a stage is complete:

resultStage.whenComplete(...
```

Prepared CQL Statements

Use executeAsync!

```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

gRPC

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

{ REST : API }

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

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

https://github.com/datastaxdevs/workshop-cassandra-application-development

Going forward

Clone the repo!

Github

https://github.com/datastaxdevs/workshop-cassandra-application-development

Have a look at the source and the initialize.cql

Feedback & Rewards

Feedback form: dtsx.io/cday-wakanda

We will deliver to your email:

★ $300 free credits code to use on Astra DB (i.e. Cassandra-as-a-Service)

★ *Apache Cassandra™ Certification Exam* Voucher (Regular Price $145)

Stay in touch!

Discord: dtsx.io/discord

Academy: academy.datastax.com

Workshops: datastax.com/workshops

YouTube: @DataStax Developers

Get Cassandra help

(*):

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

Agenda

Santa Clara

12:00 - 1:00 pm	*Check-in for Hands-on Workshop + Lunch*	
1:00 - 3:00 pm	**Hands-on Workshop: Building data-driven applications with NoSQL and Apache Cassandra®**	
3:00 - 3:30 pm	*Meetup check-in + Snacks and drinks*	
3:30 - 4:00 pm	**Intel's contributions to a faster Cassandra**	**| Smita Kamath, Shylaja Kokoori**
4:00 - 4:30 pm	**Apache Cassandra® in 2022: What to Expect from 4.1 and Beyond**	**| Scott Andreas**
4:30 - 5:00 pm	**Cassandra Performance Tuning, Tricks, and Tools**	**| Jon Haddad**
5:00 - 5:30 pm	**ACID transactions in Apache Cassandra®**	**| Patrick McFadin**
5:30 - 6:00 pm	*Food & Networking*	
6:00 - 7:00 pm	*Travel to AMC Mercado 20 - 3111 Mission College Blvd, Santa Clara*	
7:00 - 10:00 pm	**Black Panther: Wakanda Forever**	

THANK YOU!