

# Week 1

Getting Started with Cassandra



## Cassandra Workshop Series

Your path to becoming a Cassandra expert!

# The Crew





# Your hosts



Bettina Swynnerton

Community Engineer



Cédrick Lunven

Developer Advocate



David Jones-Gilardi

Developer Advocate



Aleksandr Volochnev

Developer Advocate



Jack Fryer

Community Manager





# Your hosts



Aleksandr Volochnev   
Developer Advocate



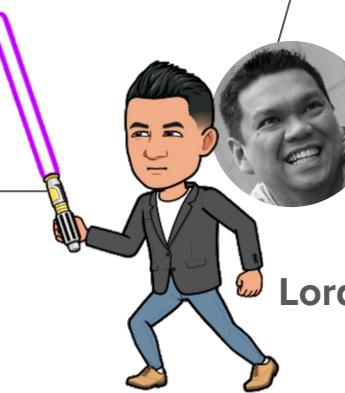
Bettina Swynnerton   
Community Engineer



Jack Fryer   
Community Manager



Cédrick Lunven   
Developer Advocate



Erick Ramirez   
Lord of the Cassandra Rings





# Developer Workshop Series

**What we will  
cover:**

- Bootstrapping
- Apache Cassandra™ Why, What & When
- Read and Write path
- Uber High Level Data Modeling
- What's NEXT?

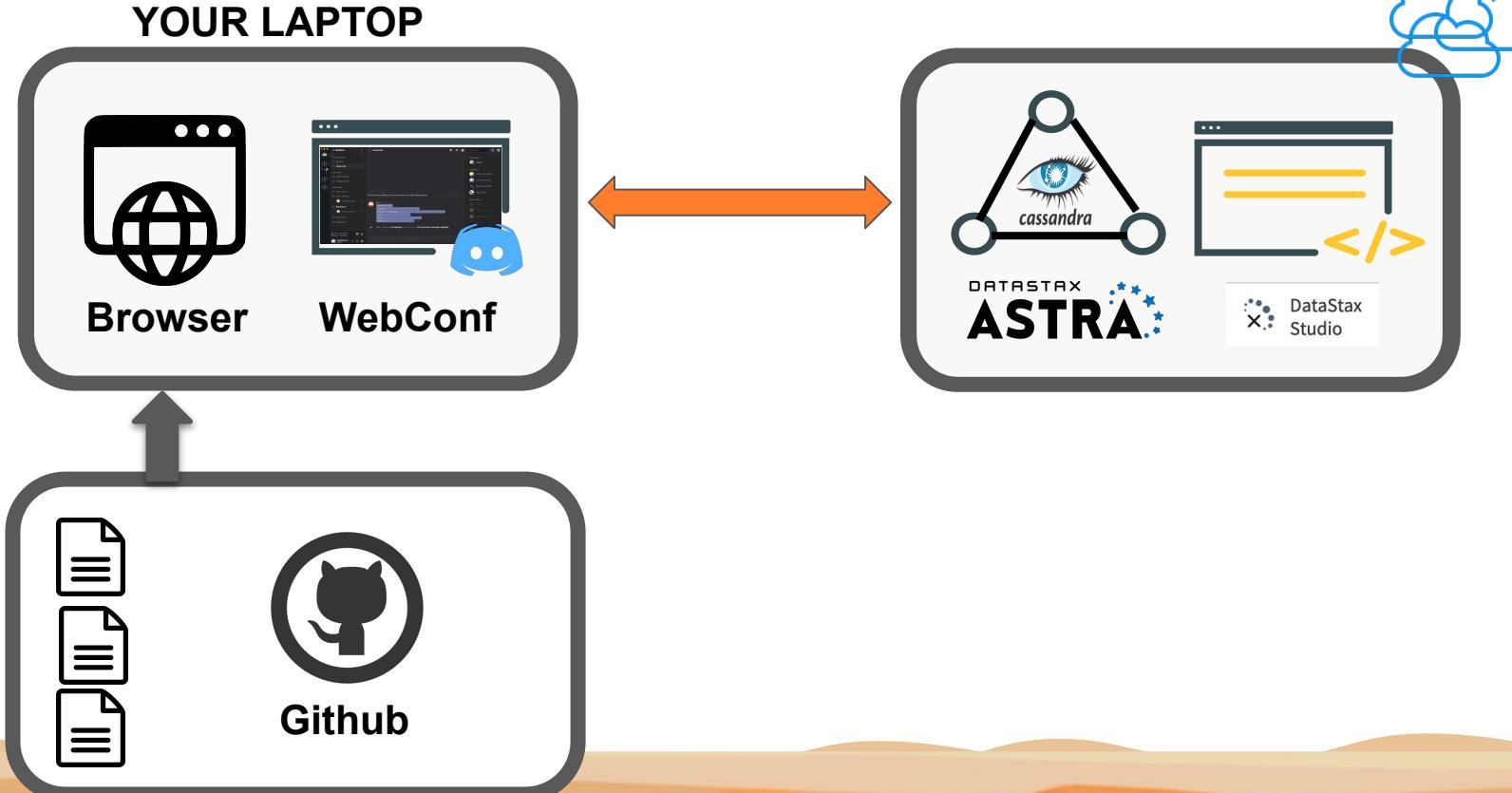


# Developer Workshop Series

**What we will  
cover:**

- Bootstrapping
- Apache Cassandra™ Why, What & When
- Read and Write path
- Uber High Level Data Modeling
- What's NEXT?

# Overview



# LiveStreams

**Cassandra Developer Workshop**

**3 HOURS**

- 1 Housekeeping and Quizz
- 2 Apache Cassandra™ Why, What & Who
- 3 Data Modeling with Apache Cassan
- 4 Advanced Data Types
- 5 LightWeight Transactions and Batch
- 6 Resources

**YouTube**

**Twitch**

DRTV

## Questions during livestreams

**community.datastax.com**

**astra.datastax.com**

The screenshot shows the DataStax Astra Cloud interface. At the top, there's a navigation bar with 'DataStax Astra' logo, 'Cedrick's Databases', and a dropdown menu. Below it, a '1 Database' summary is shown. The main area has a breadcrumb path: 'Astra Databases > DevWorkshopDb'. It features several tabs: 'Actions' (selected), 'Status' (Active), 'Summary', 'COL Console', and 'Health'. On the left, there's a sidebar with a cluster overview and coordinator metrics. The main content area displays various metrics for the cluster, including requests served per second, latency, and memtable space usage over time.



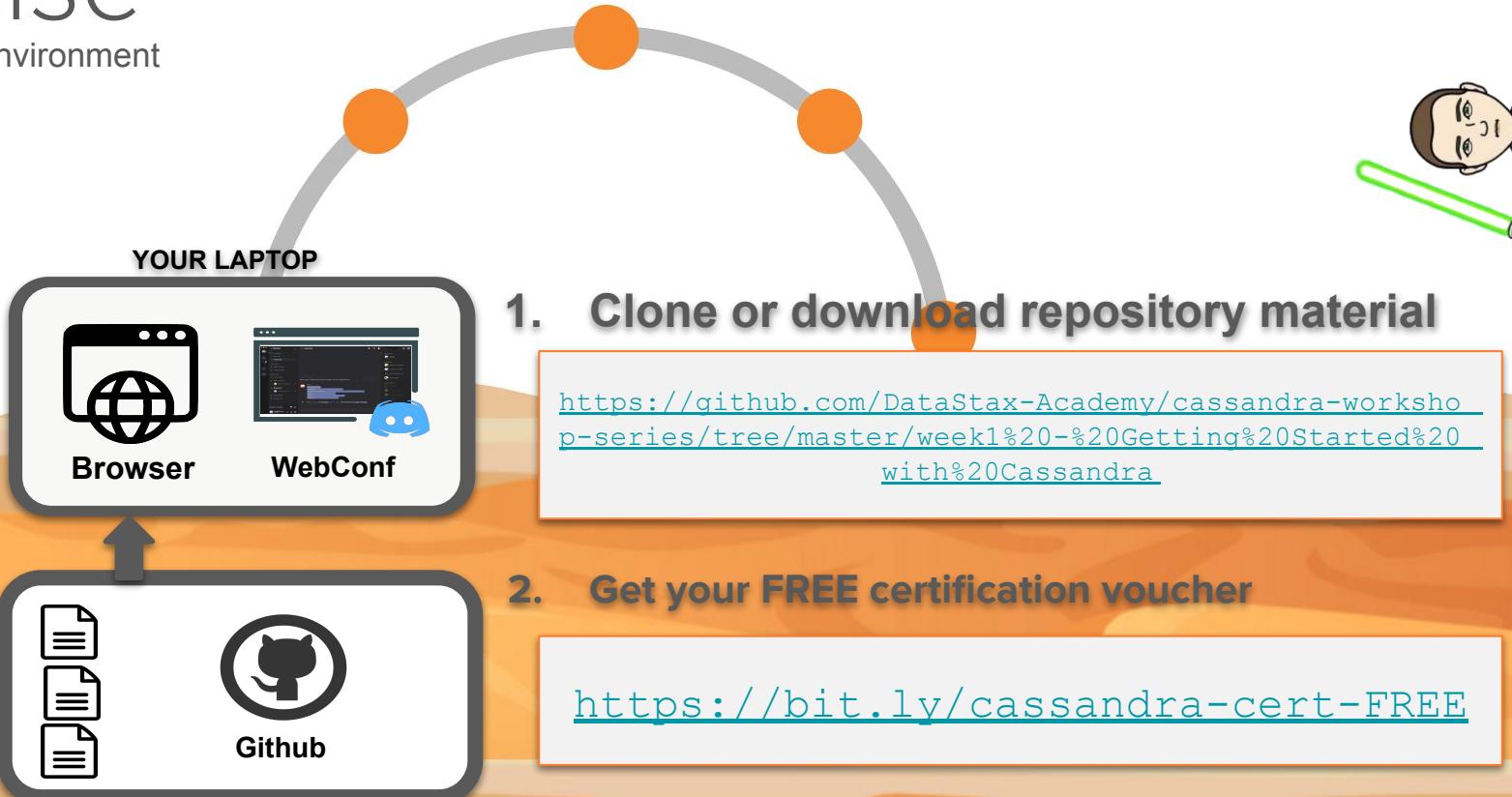
The screenshot shows a GitHub repository page for 'DataStax-Academy / cassandra-workshop-series'. The repository has 6 commits, 1 branch, and 0 tags. The README.md file is the latest commit. The repository is public and has 1 star.



#CassandraWorkshopSeries

# Exercise

Bootstrap your environment



# Introducing Astra



## Eliminate Operations

everything from provisioning to backups is fully automated



## Secure Your Data

with the most advanced security available for Cassandra



## Simplify App Development

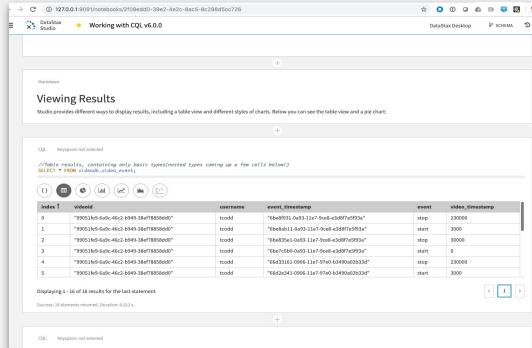
with auto-configured developer tools that deploy with a click

# Simplify Application Development

## Familiar Language

```
INSERT INTO mytable
(id, name, address) VALUES
(1, 'Bob Smith', '1 Main
Street')
SELECT * FROM mytable
WHERE id=1
UPDATE mytable SET
name='Tom Smith' WHERE
id=1
DELETE FROM mytable WHERE
id=1
```

## Easy Dev Tools



The screenshot shows the DataStax Studio interface with a CQL query results table titled "Viewing Results". The table has columns: index, eventid, timestamp, event, and video\_id. The data is as follows:

index	eventid	timestamp	event	video_id
0	"99999999-0000-4000-8000-340478785600"	"1e4eff00-0a93-11e7-9e0d-a5287f1c7294"	stop	33000
1	"99999999-0000-4000-8000-340478785600"	"1e4ebab1-0a93-11e7-9e0d-a5287f1c7294"	start	3000
2	"99999999-0000-4000-8000-340478785600"	"1e4ec5a1-0a93-11e7-9e0d-a5287f1c7294"	stop	30000
3	"99999999-0000-4000-8000-340478785600"	"1e4ef28b-0a93-11e7-9e0d-a5287f1c7294"	start	3
4	"99999999-0000-4000-8000-340478785600"	"1e4ef311-0a93-11e7-9e0d-a5287f1c7294"	stop	330000
5	"99999999-0000-4000-8000-340478785600"	"1e4ef344-0a93-11e7-9e0d-a5287f1c7294"	start	3000

## Great Drivers

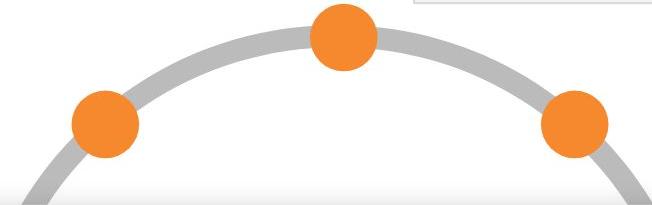


# Exercise

## Creating your Astra instance



<https://github.com/DataStax-Academy/cassandra-workshop-series/tree/master/week1%20-%20Getting%20Started%20with%20Cassandra#1-create-your-astra-instance>



The screenshot shows the DataStax Constellation interface with the following details:

- Summary** tab selected.
- Databases**: 2 Databases
- Message**: Database is initializing. You'll receive an email when the database is ready. While you're waiting, learn how to [get started](#) with your database. Database creation started at Jan 22nd 2020 11:49am UTC. This can take as long as 30 minutes but is usually much quicker.
- Apollo Databases > screenshot**
- screenshot** panel:
  - Keyspace Name: okidoki
  - Organization: Your Organization
  - Owner: Cedrick Lunven
  - Created: January 22, 2020
- Size and Location** panel:
  - Capacity Unit: 1
  - Total Storage: 500 GB
  - Storage Used: (empty)
  - Locations: us-east-1 (1 capacity unit)
  - Compute Size: Startup
  - Replication Factor: 3
- Cost** panel:
  - Spent this month: TBD
  - Estimated Cost**
    - Per Hour
    - Per Month
  - when running: TBD/hour
  - when parked: TBD/hour
- Connection Details** panel:

Connection details are only available for active databases that you own or have connection permissions for.



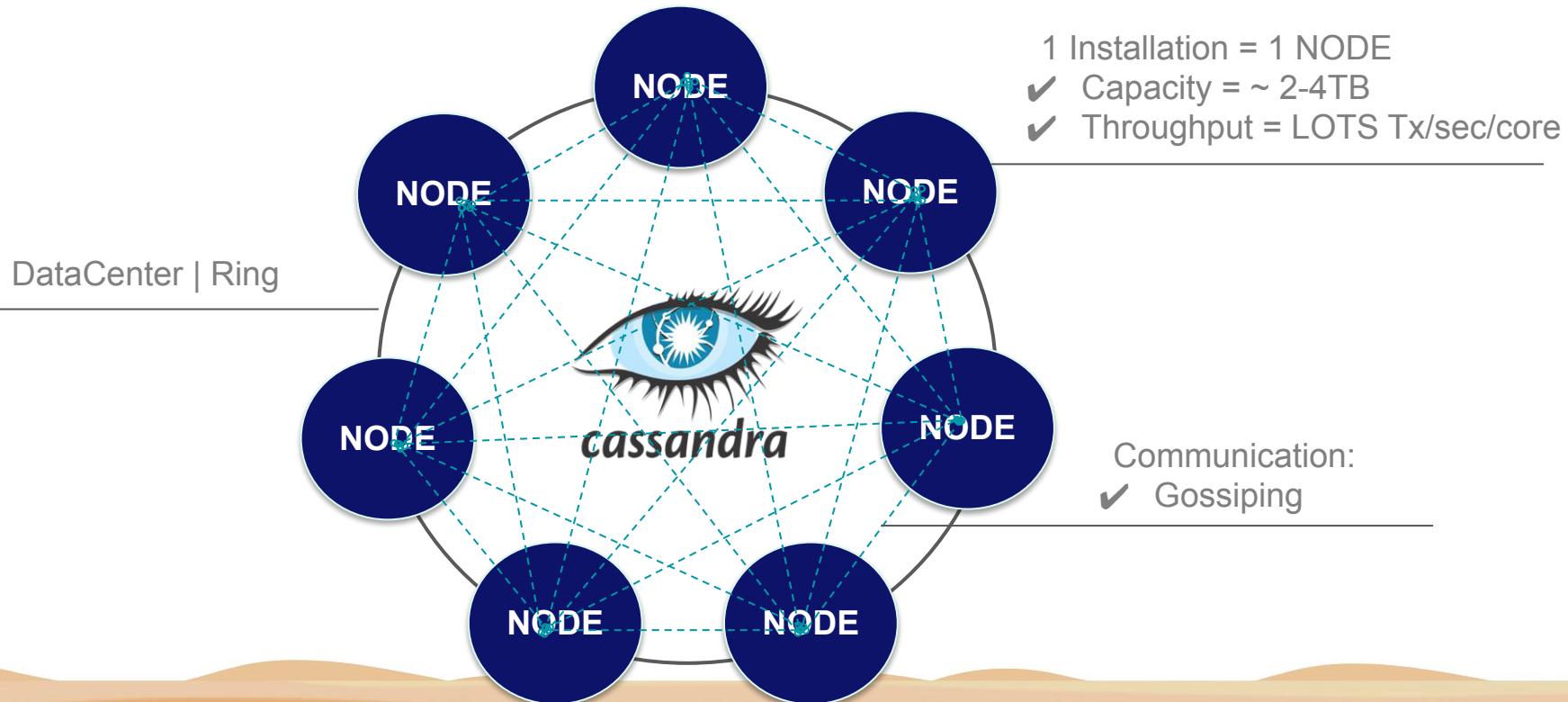


# Developer Workshop Series

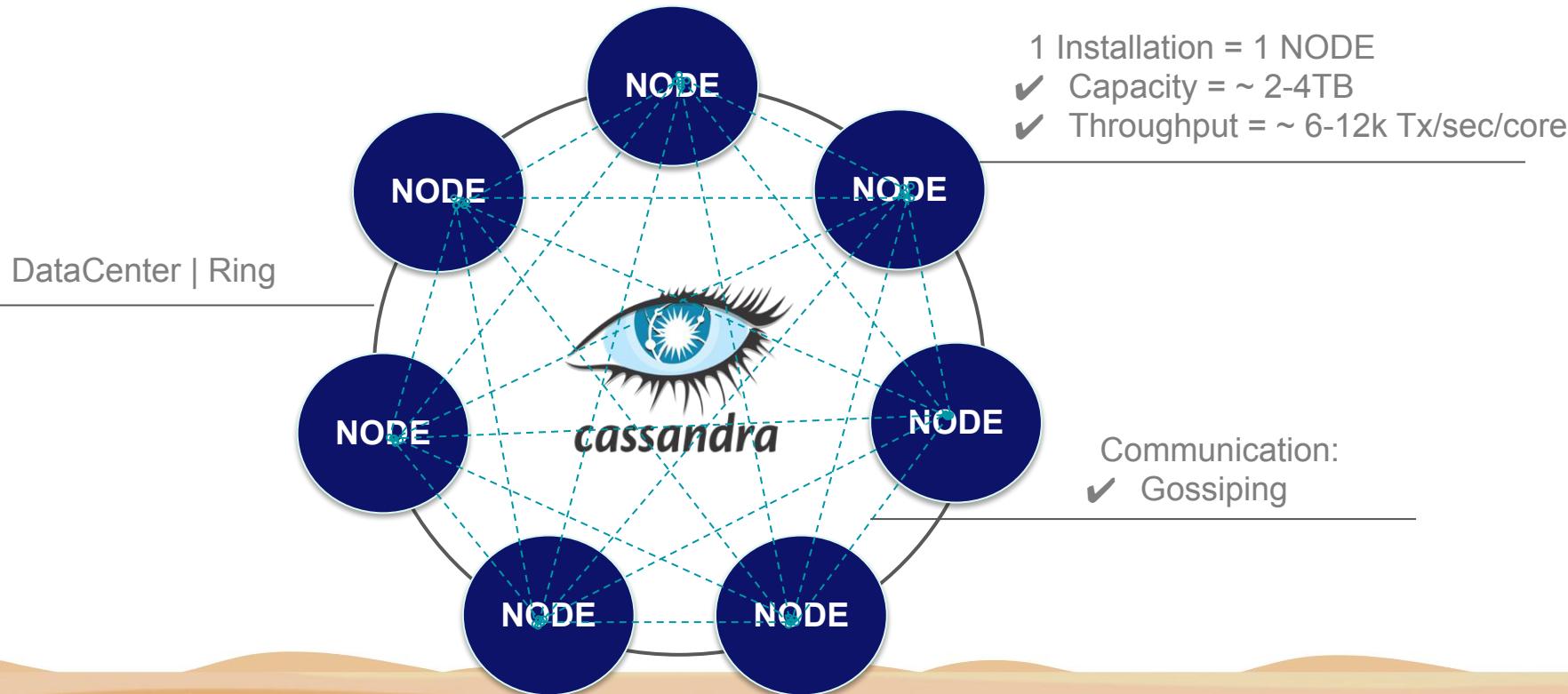
**What we will  
cover:**

- Bootstrapping
- Apache Cassandra™ Why, What & When
- Read and Write path
- Uber High Level Data Modeling
- What's NEXT?

# Apache Cassandra™ = NoSQL Distributed Database



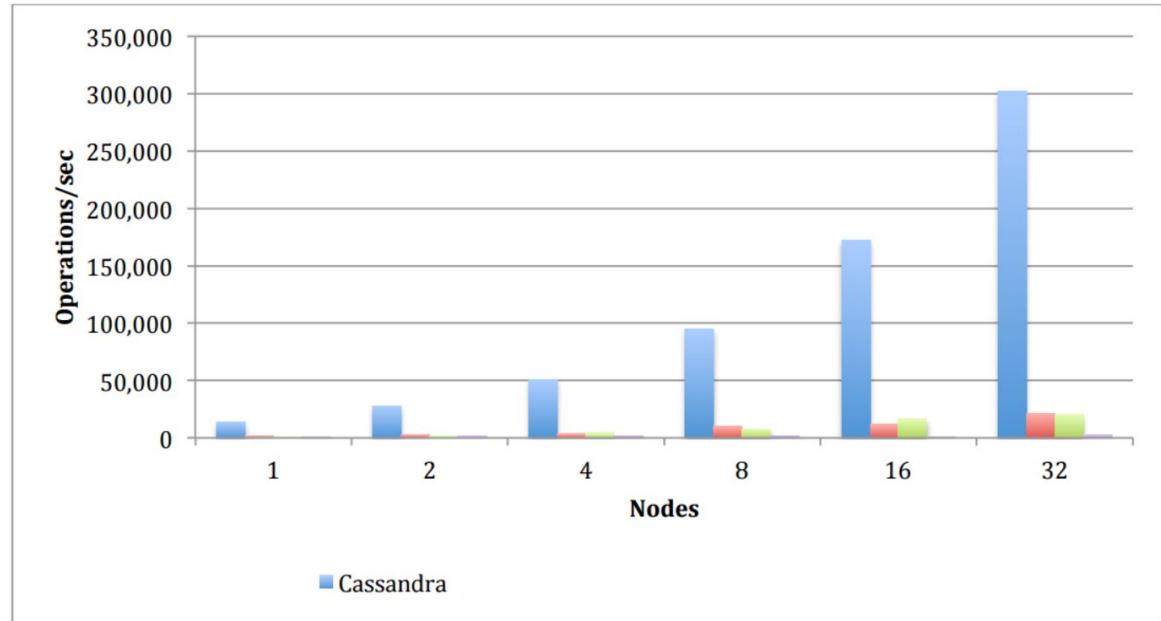
# Apache Cassandra™ = NoSQL Distributed Database



# Scales Linearly

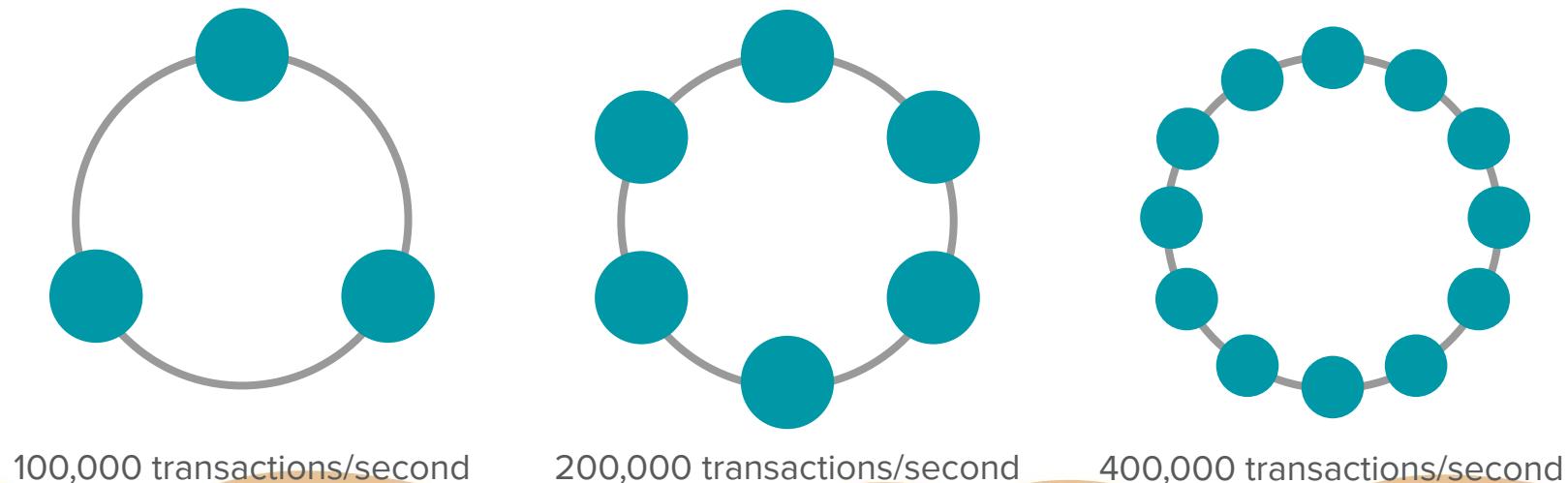
- Need more capacity?
- Need more throughput?
- Add nodes!

Balanced Read/Write Mix



# Horizontal vs. Vertical Scaling

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



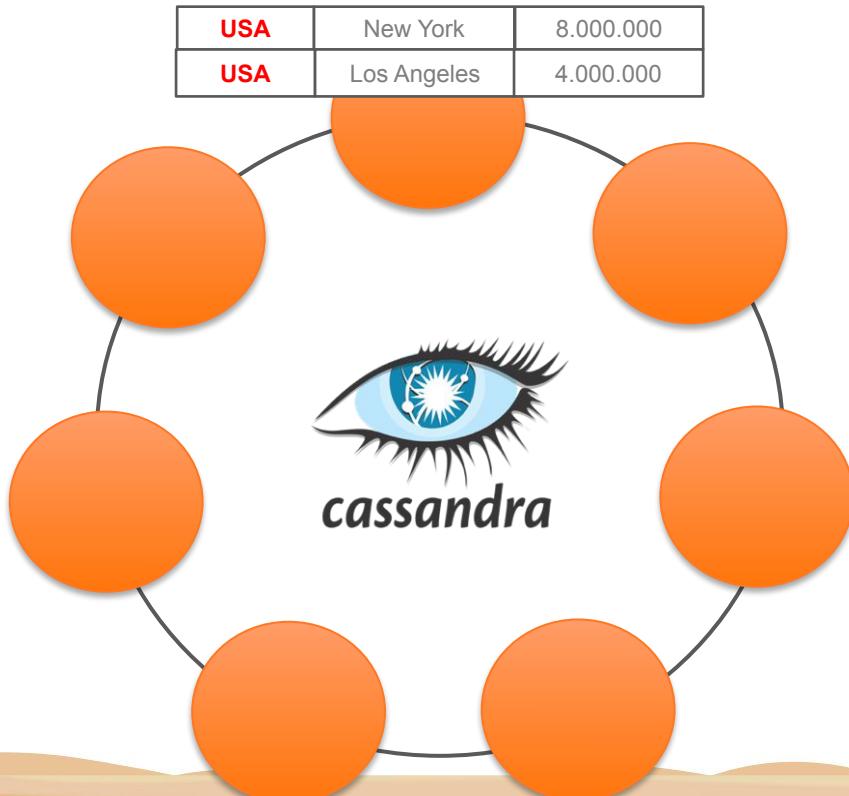
# Data is Distributed



Country	City	Population
USA	New York	8.000.000
USA	Los Angeles	4.000.000
FR	Paris	2.230.000
DE	Berlin	3.350.000
UK	London	9.200.000
AU	Sydney	4.900.000
DE	Nuremberg	500.000
CA	Toronto	6.200.000
CA	Montreal	4.200.000
FR	Toulouse	1.100.000
JP	Tokyo	37.430.000
IN	Mumbai	20.200.000

{  
    }  
Partition Key

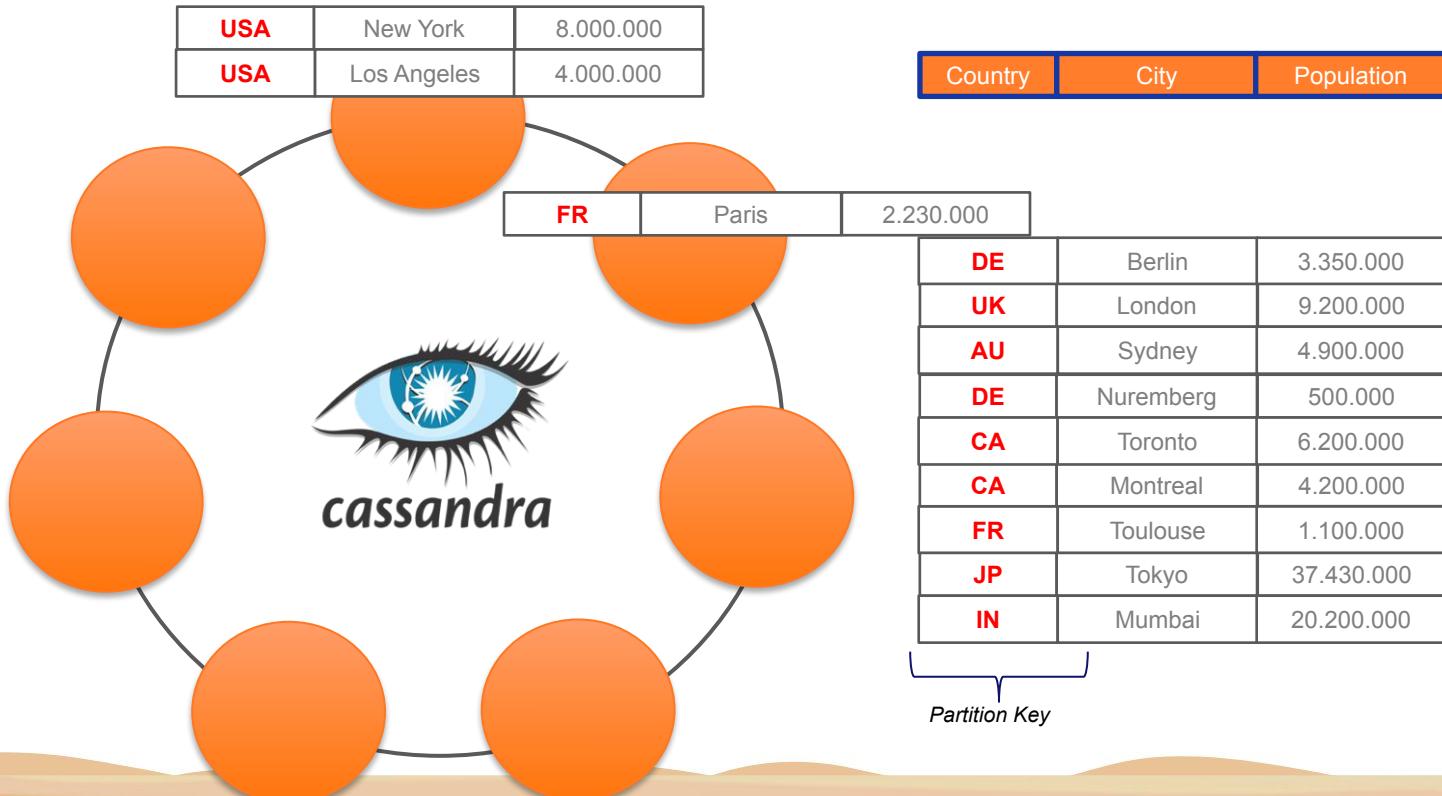
# Data is Distributed



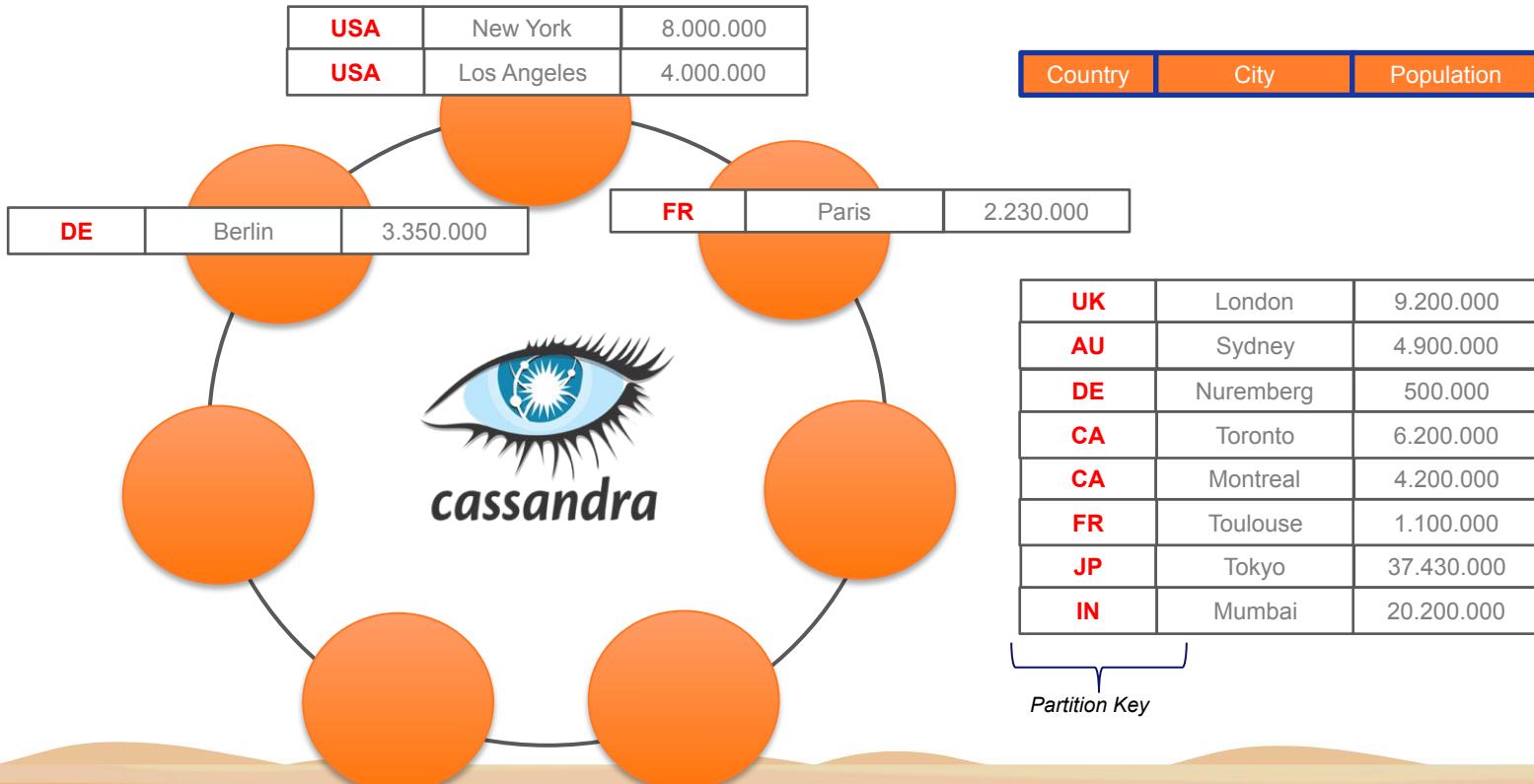
FR	Paris	2.230.000
DE	Berlin	3.350.000
UK	London	9.200.000
AU	Sydney	4.900.000
DE	Nuremberg	500.000
CA	Toronto	6.200.000
CA	Montreal	4.200.000
FR	Toulouse	1.100.000
JP	Tokyo	37.430.000
IN	Mumbai	20.200.000

Partition Key

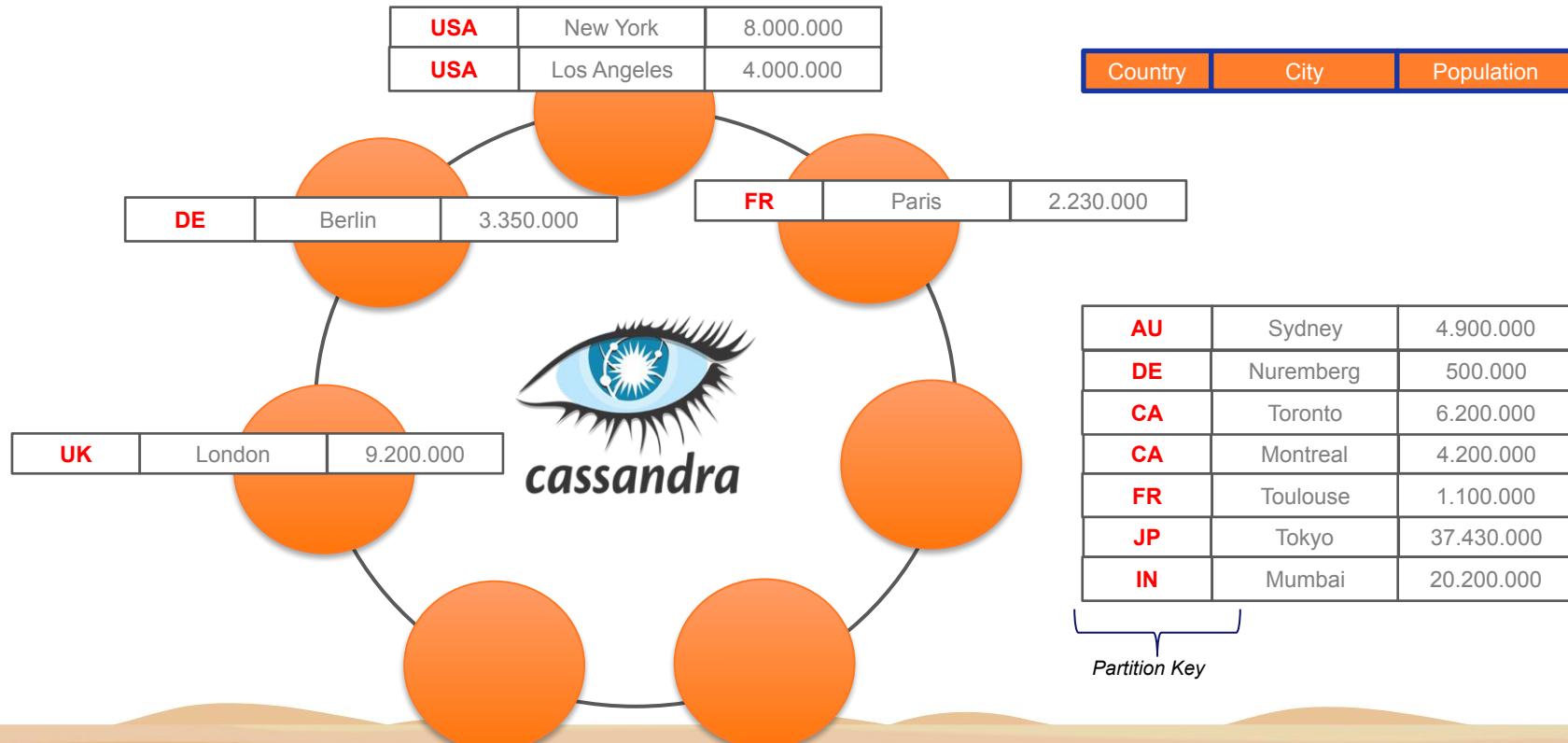
# Data is Distributed



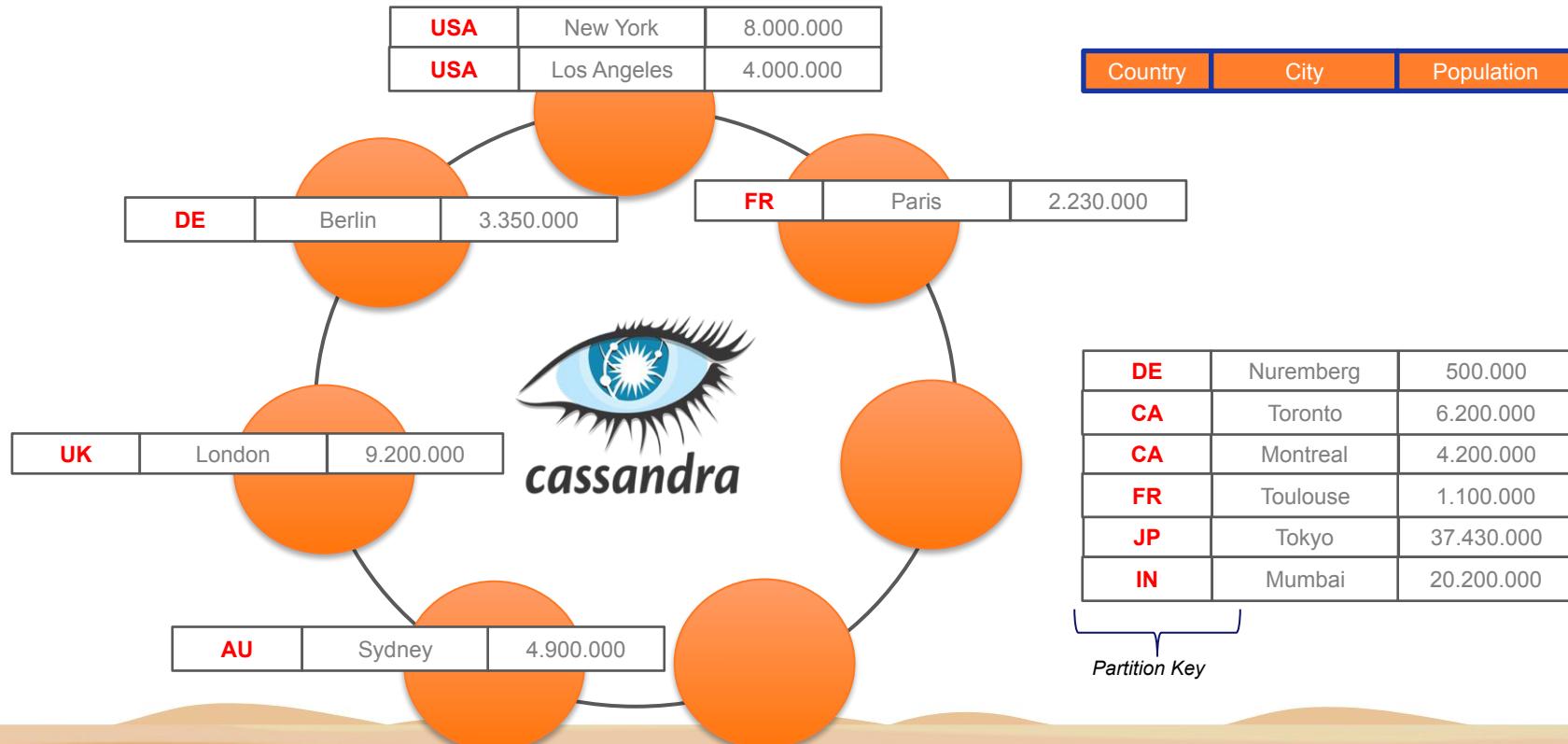
# Data is Distributed



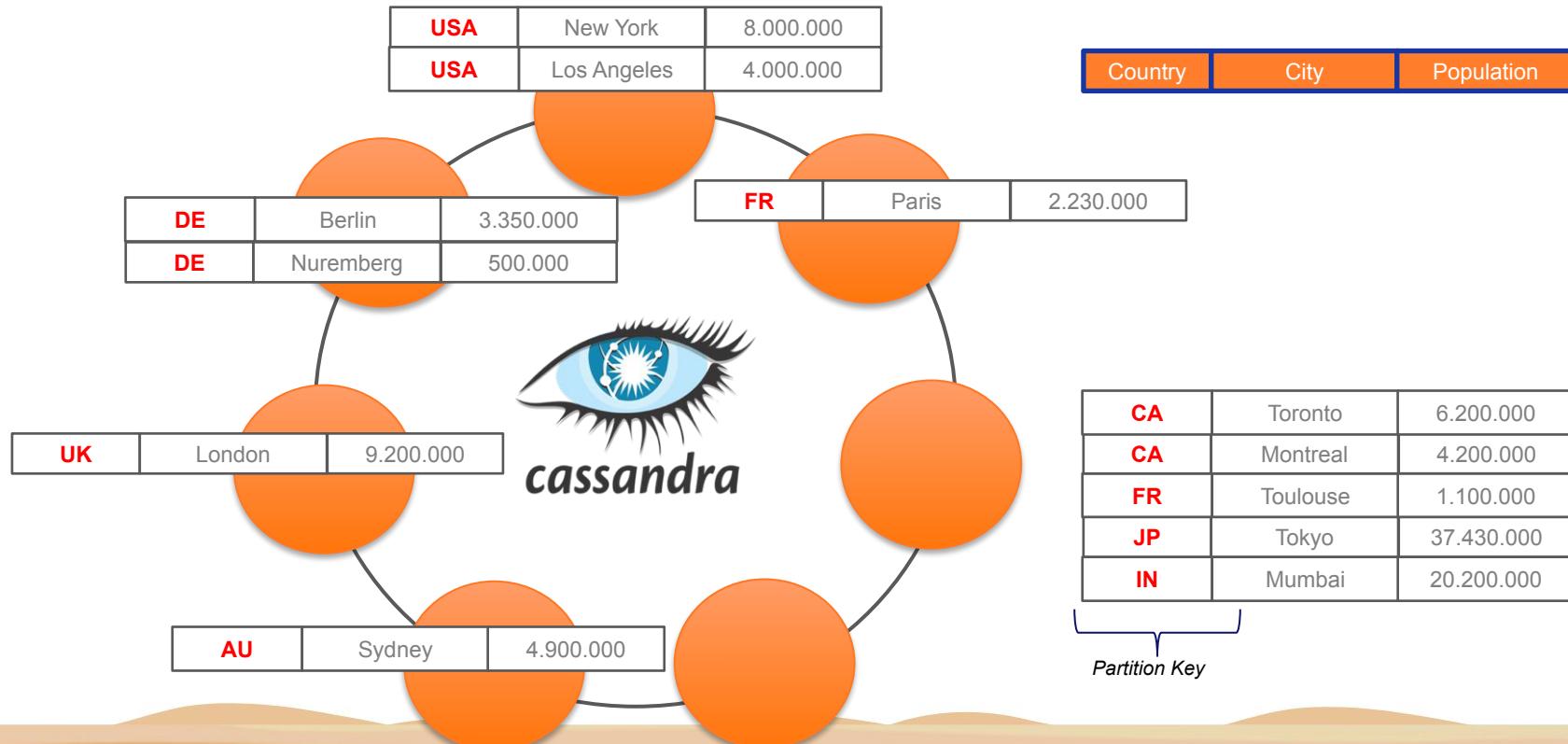
# Data is Distributed



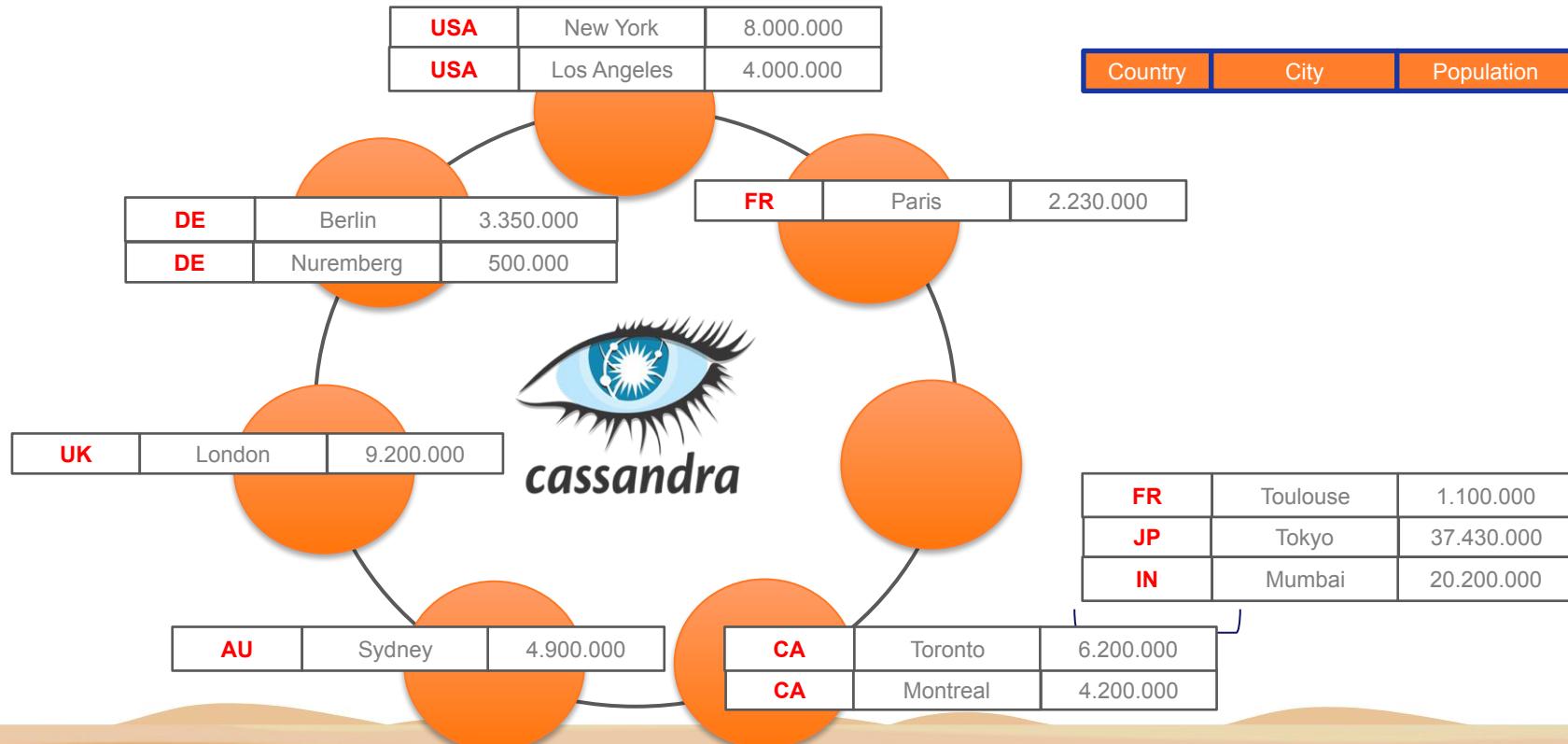
# Data is Distributed



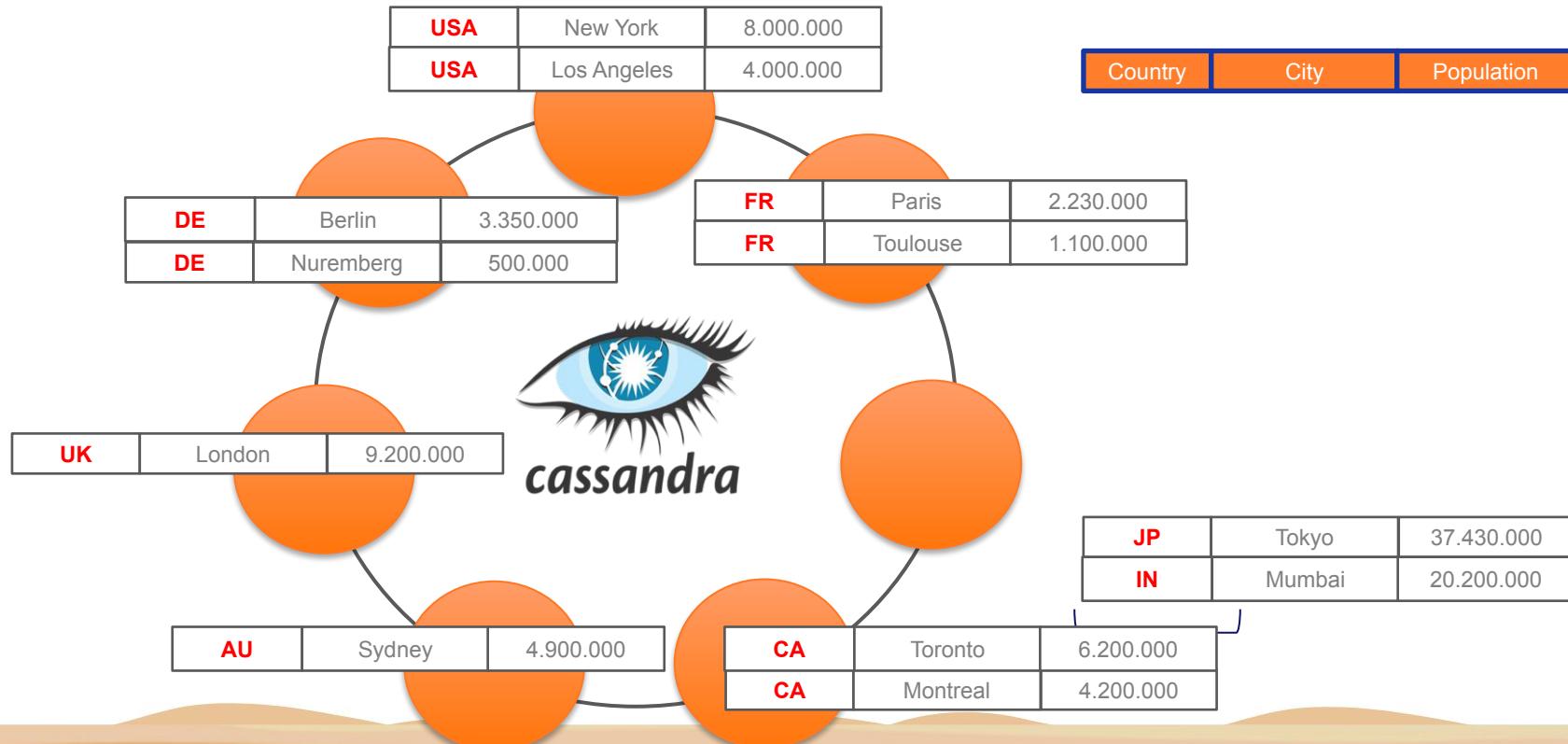
# Data is Distributed



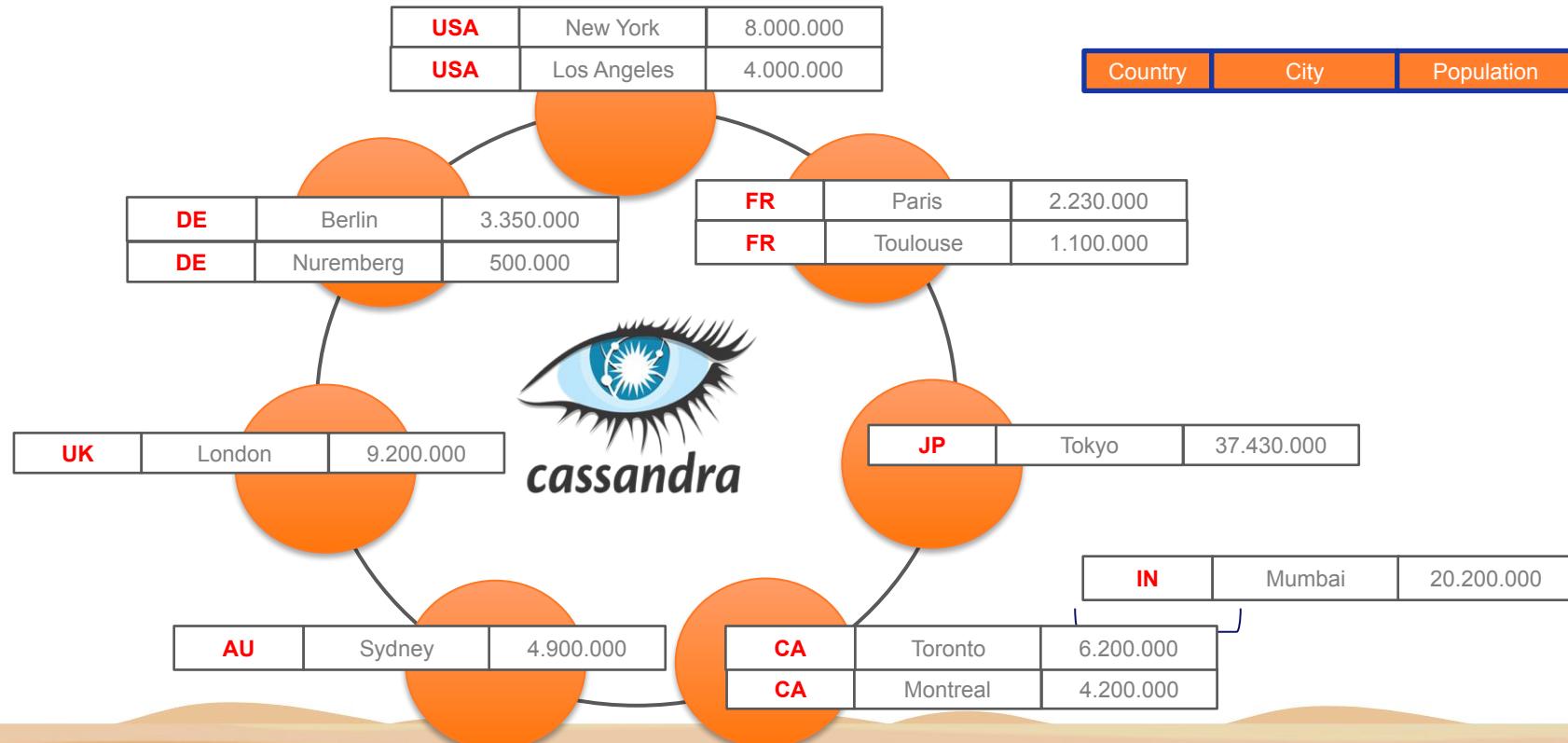
# Data is Distributed



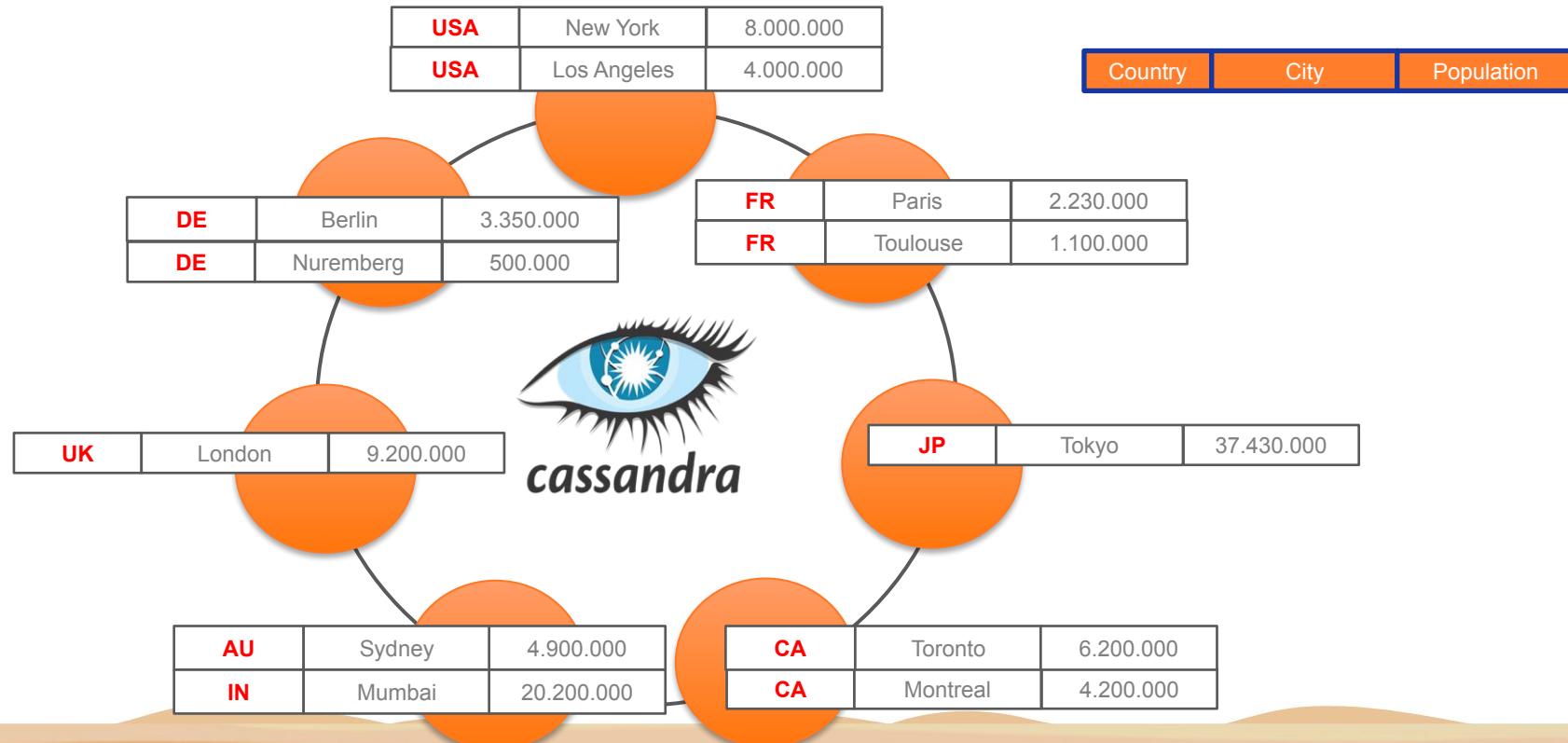
# Data is Distributed



# Data is Distributed



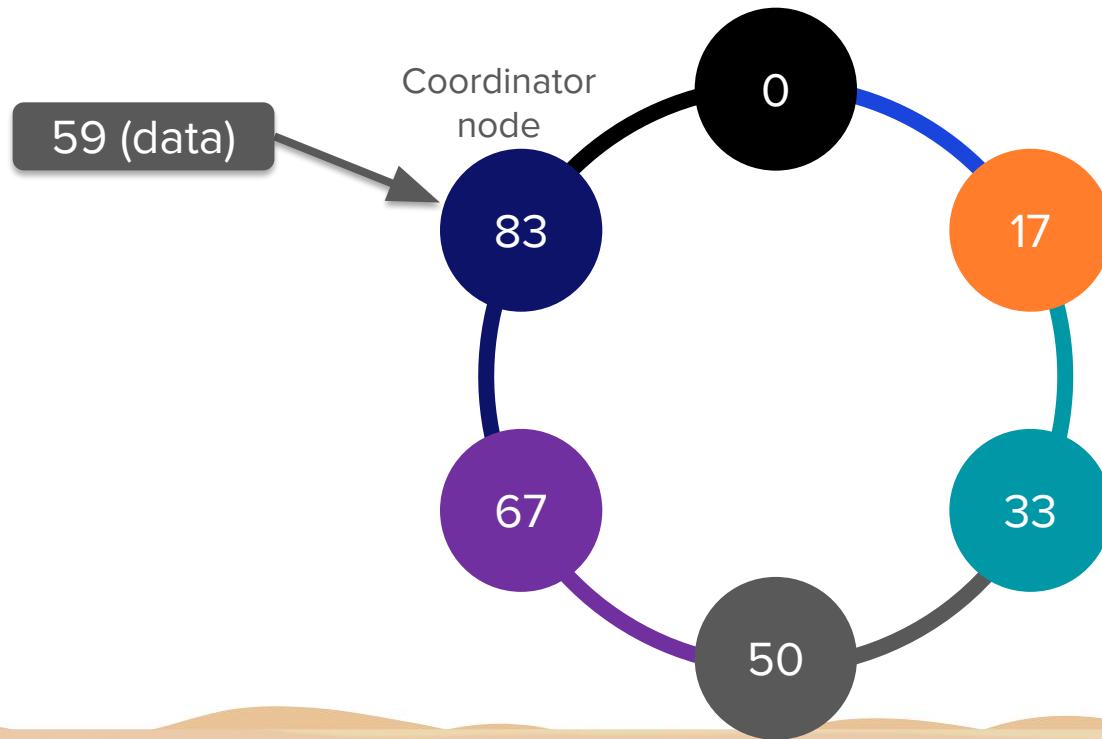
# Data is Distributed



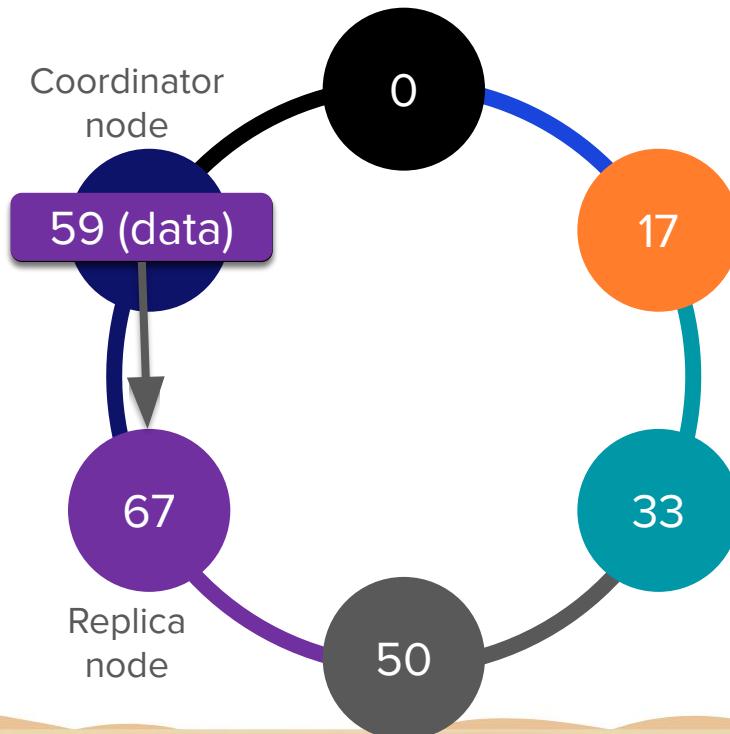
# Replication



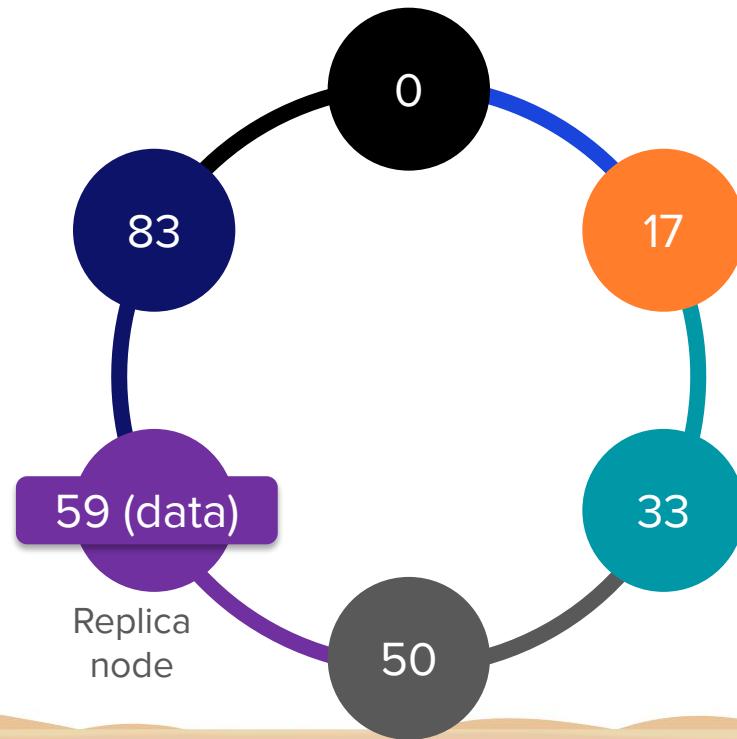
# How the Ring Works



# How the Ring Works

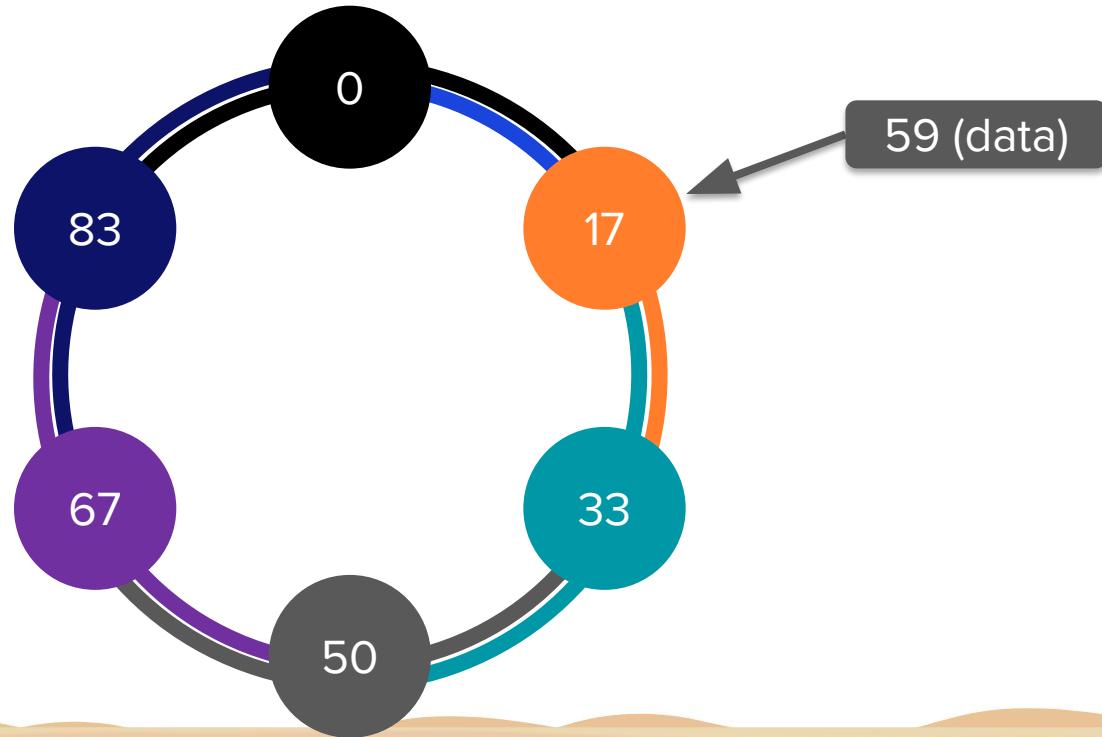


# How the Ring Works



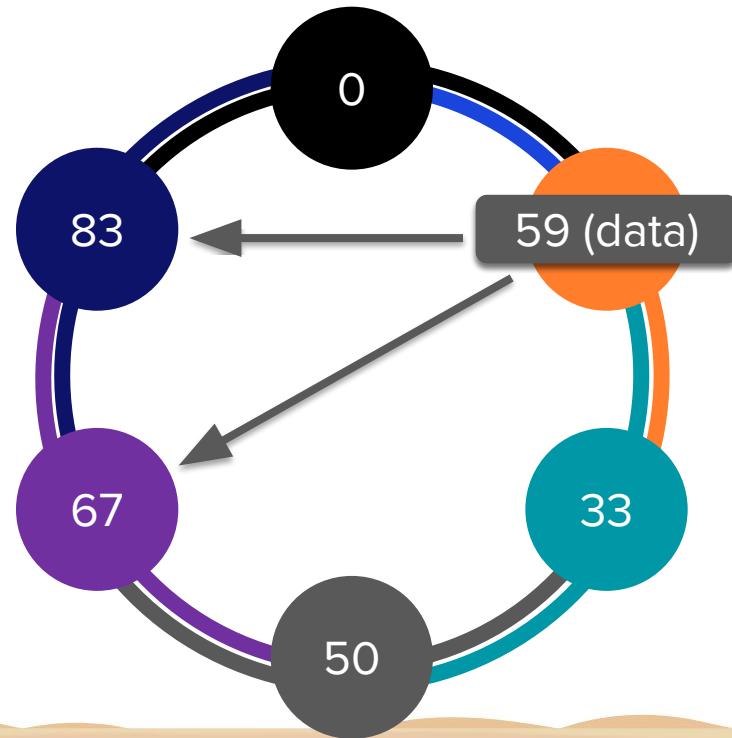
# Replication within the Ring

RF = 2



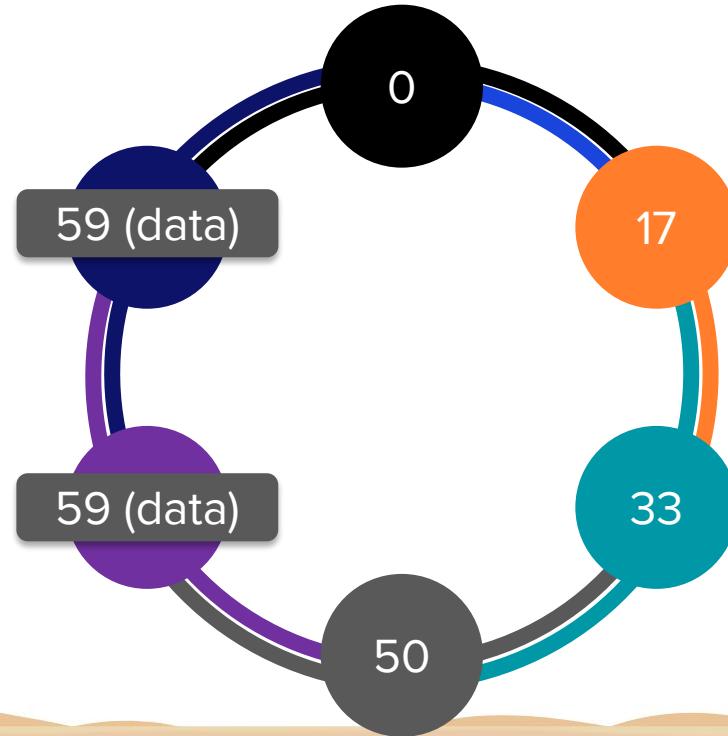
# Replication within the Ring

RF = 2



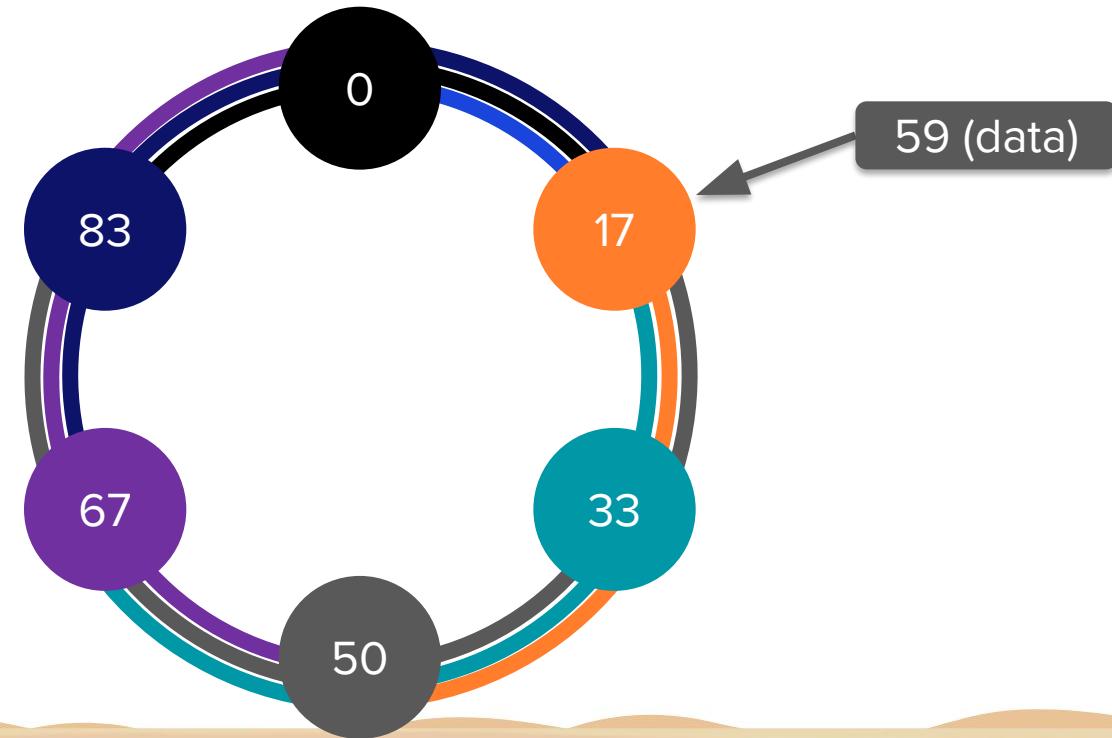
# Replication within the Ring

RF = 2



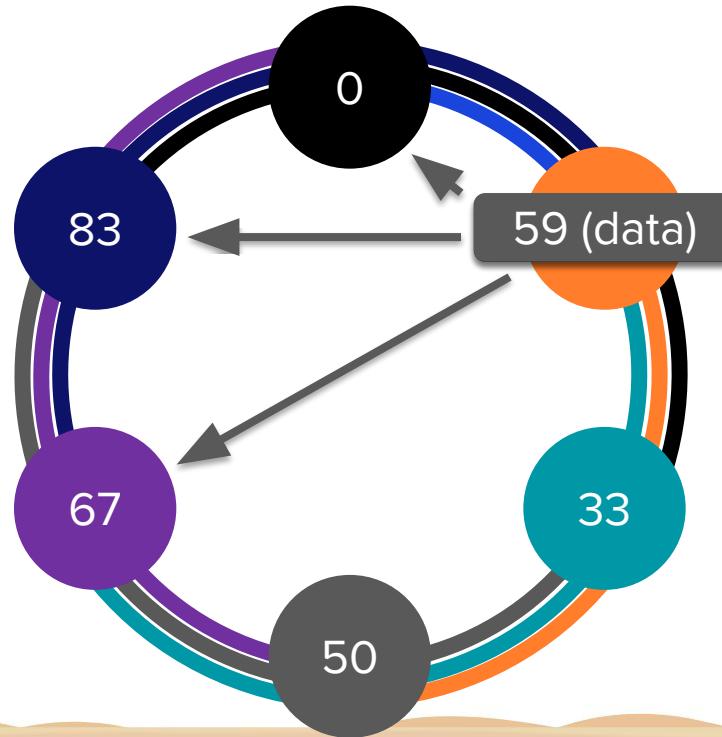
# Replication within the Ring

RF = 3



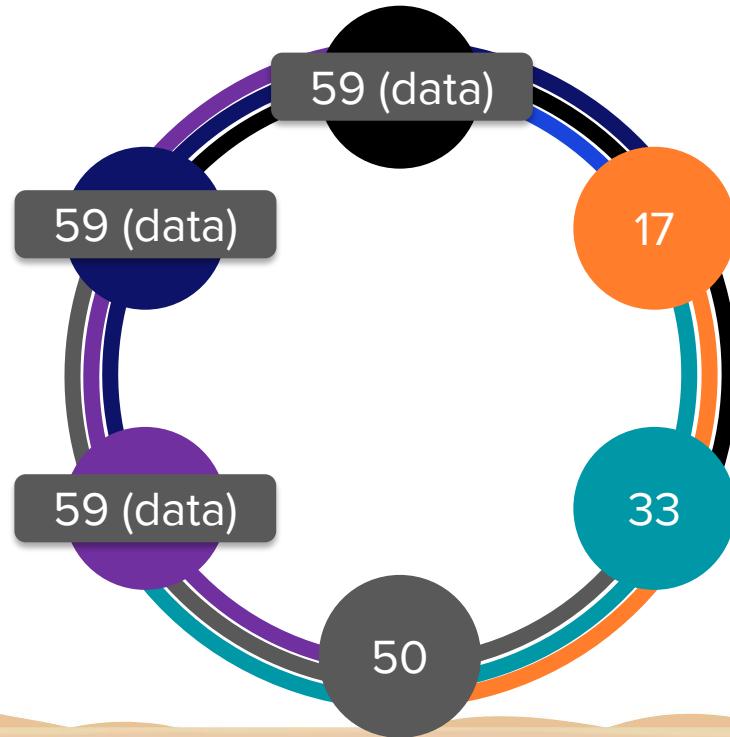
# Replication within the Ring

RF = 3



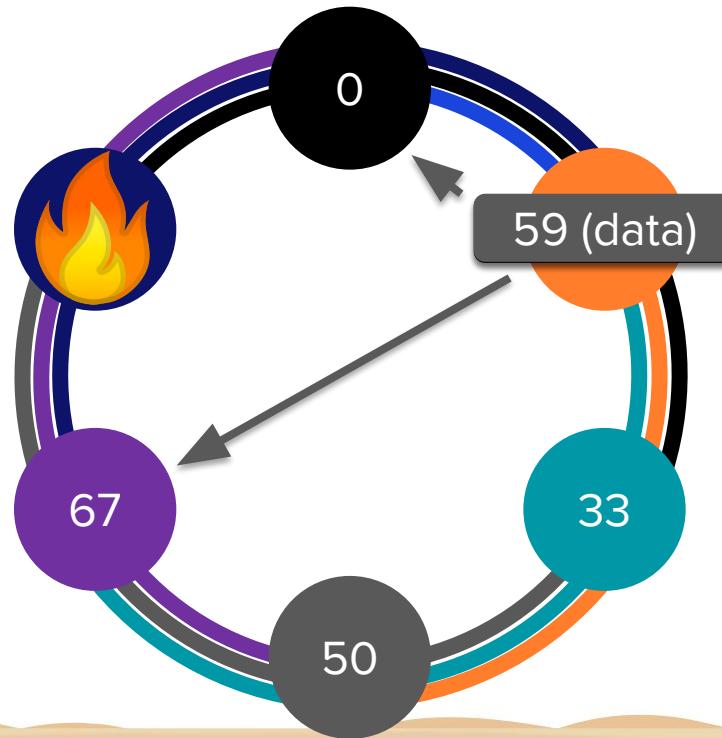
# Replication within the Ring

RF = 3



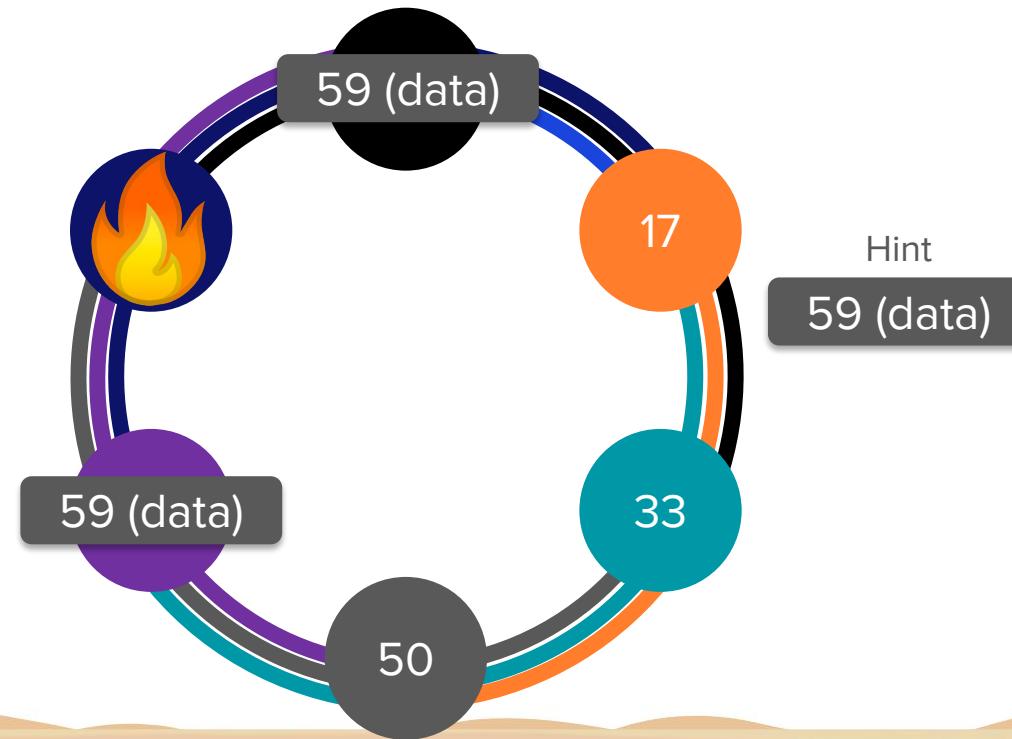
# Node Failure

RF = 3



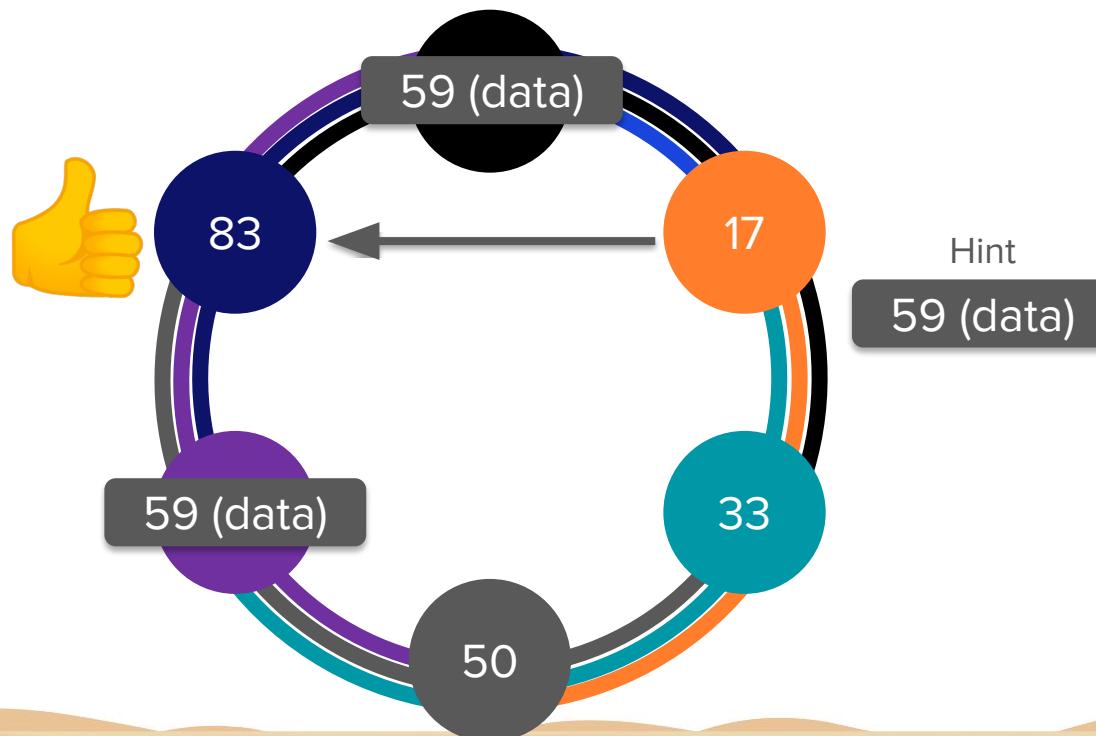
# Node Failure

RF = 3



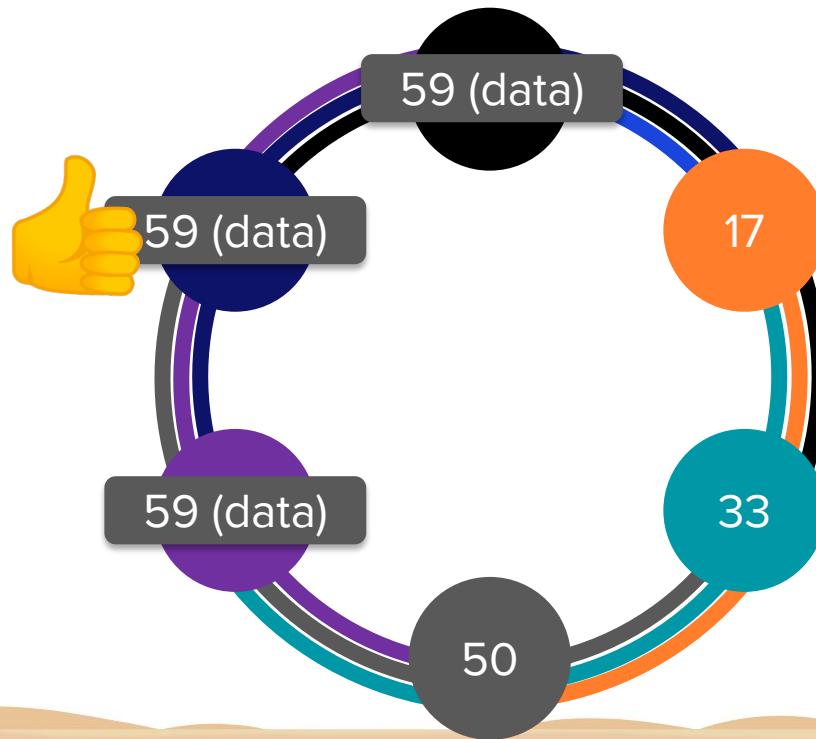
# Node Failure

RF = 3



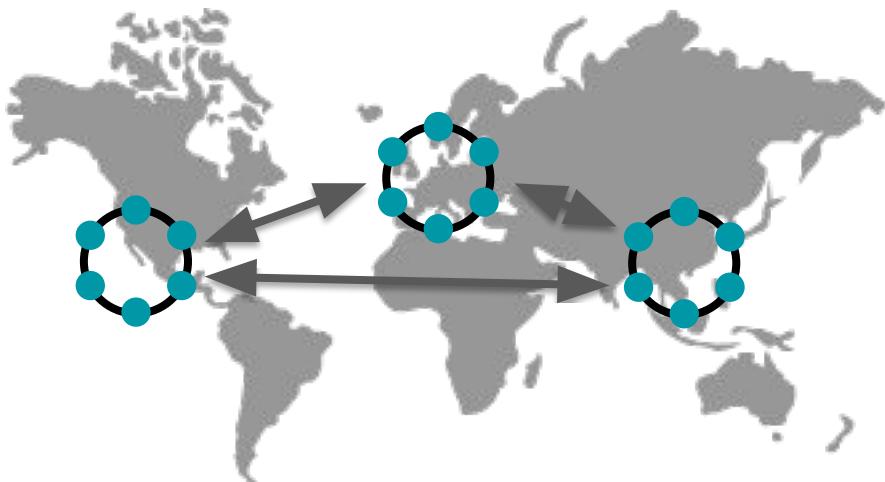
# Node Failure – Recovered!

RF = 3

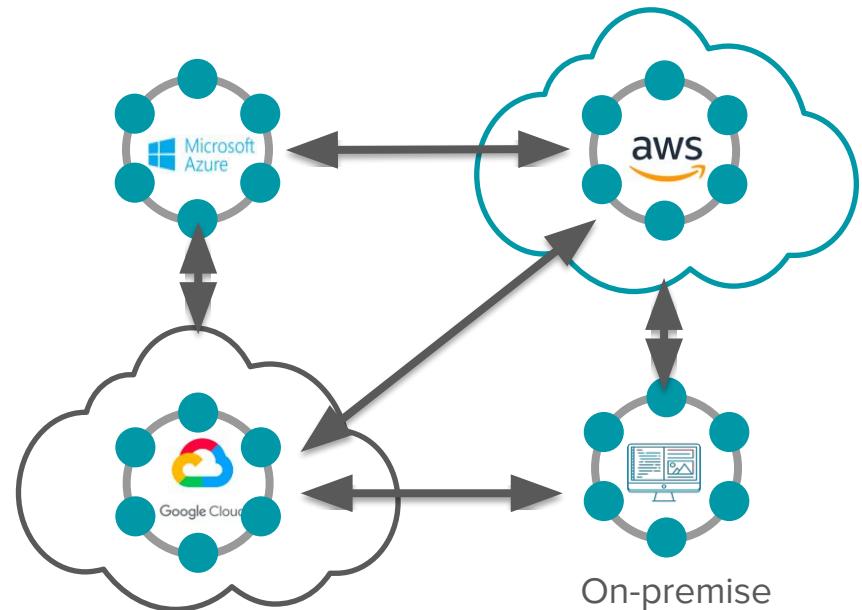


# Data Distributed Everywhere

- Geographic Distribution



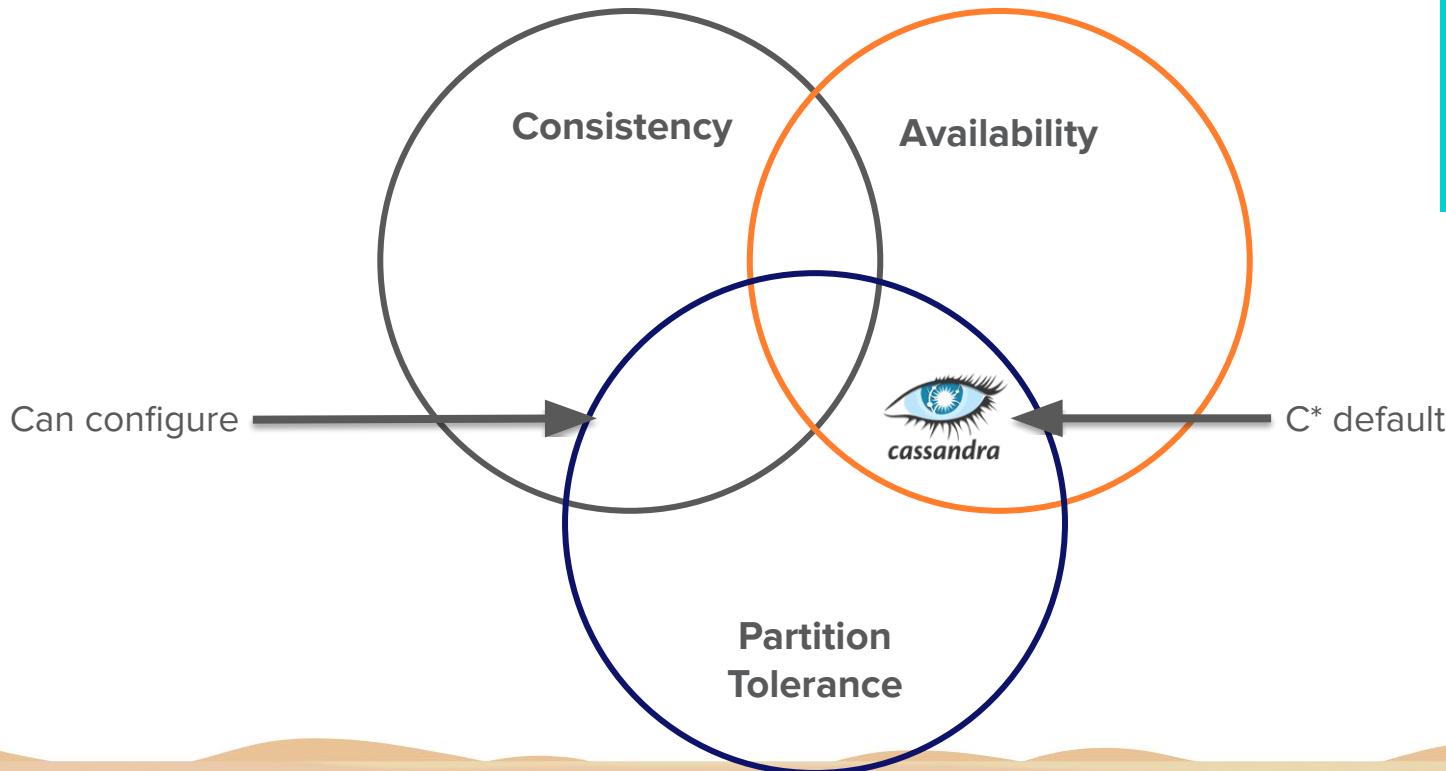
- Hybrid-Cloud and Multi-Cloud



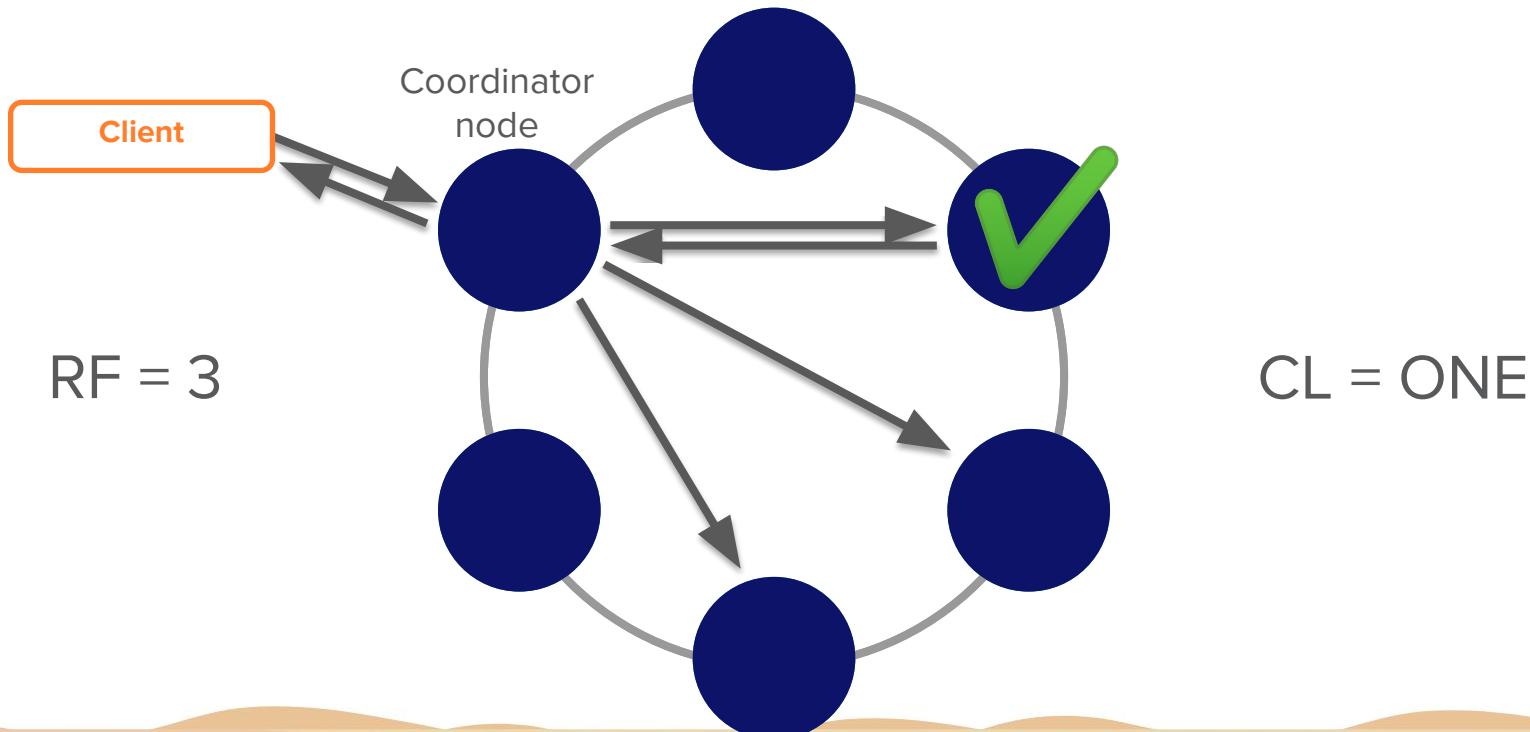
# Consistency



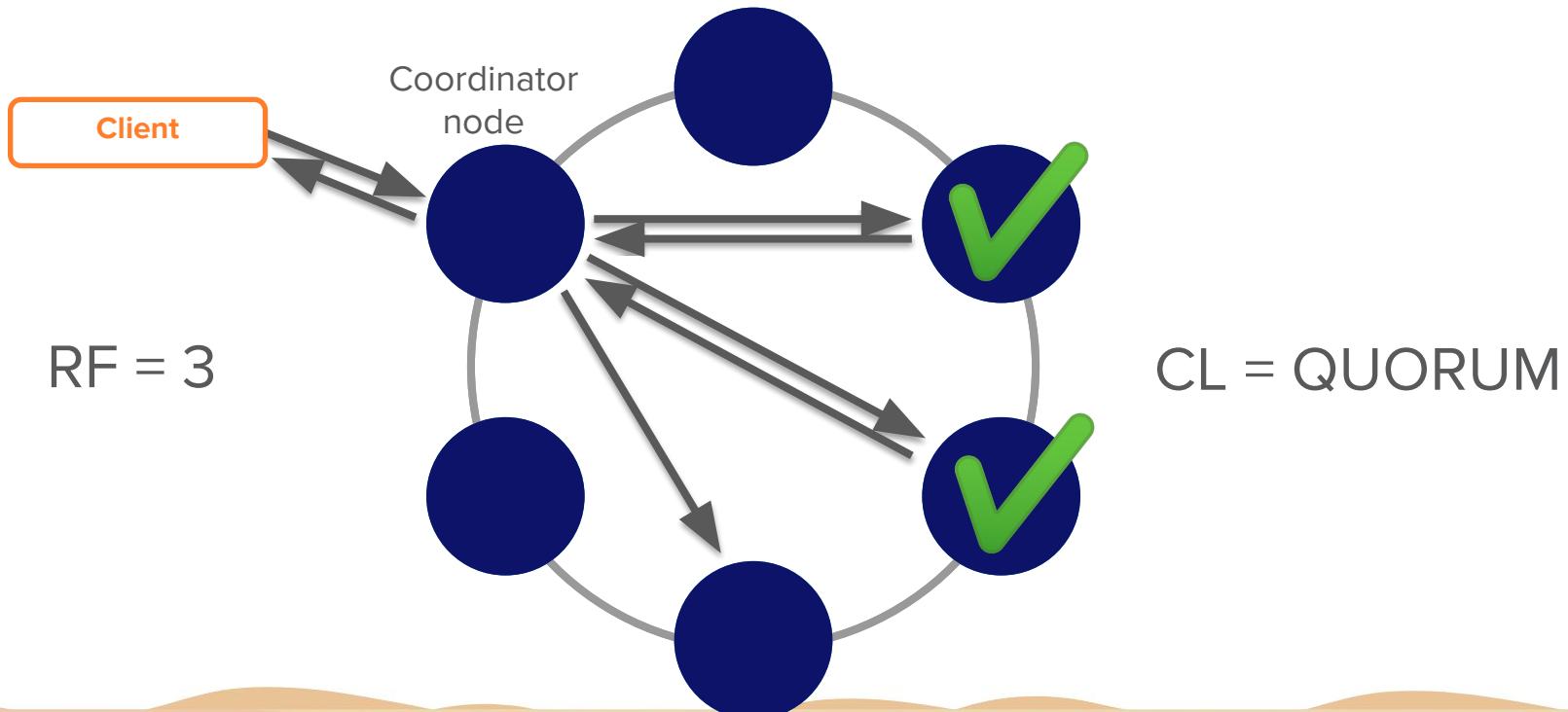
# CAP Theorem



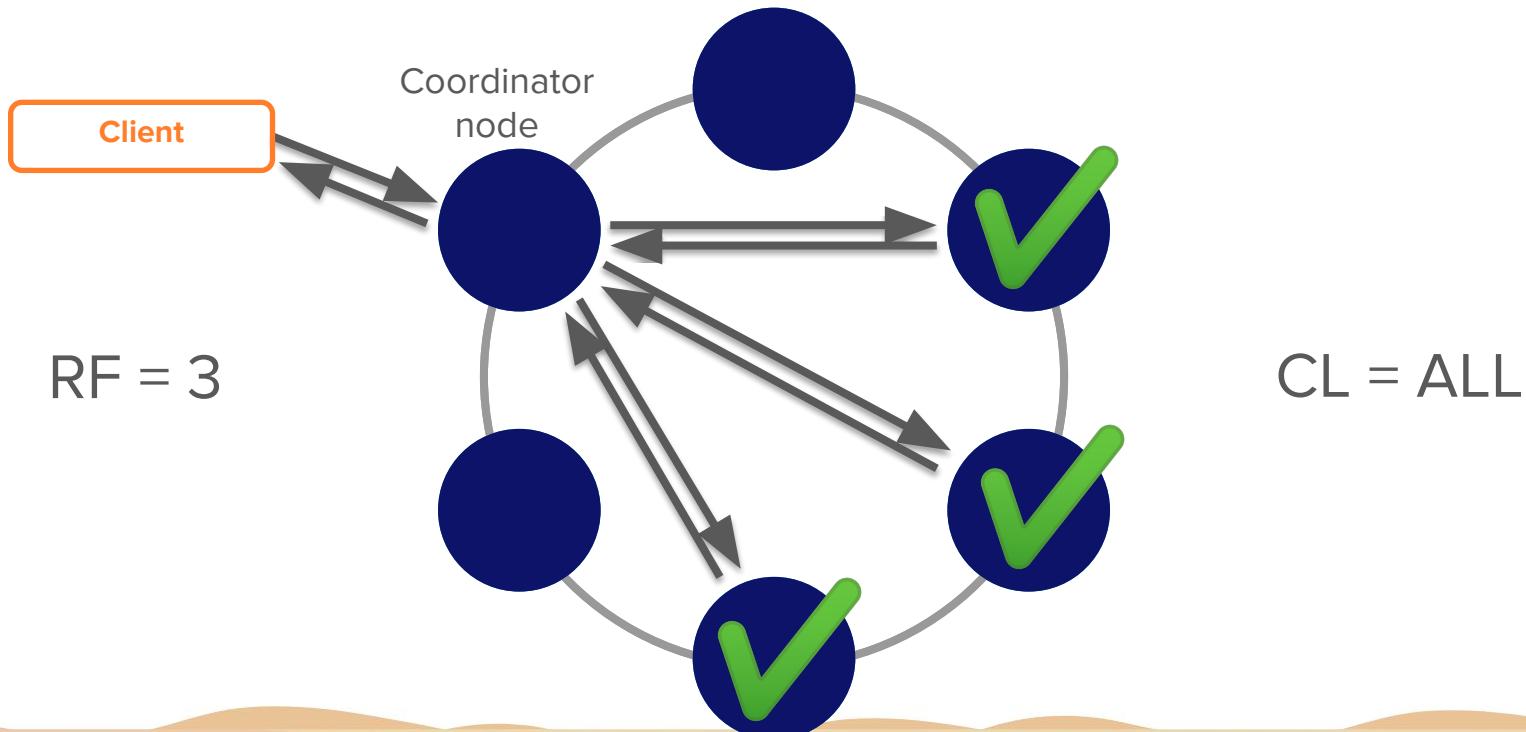
# Consistency Levels



# Consistency Levels



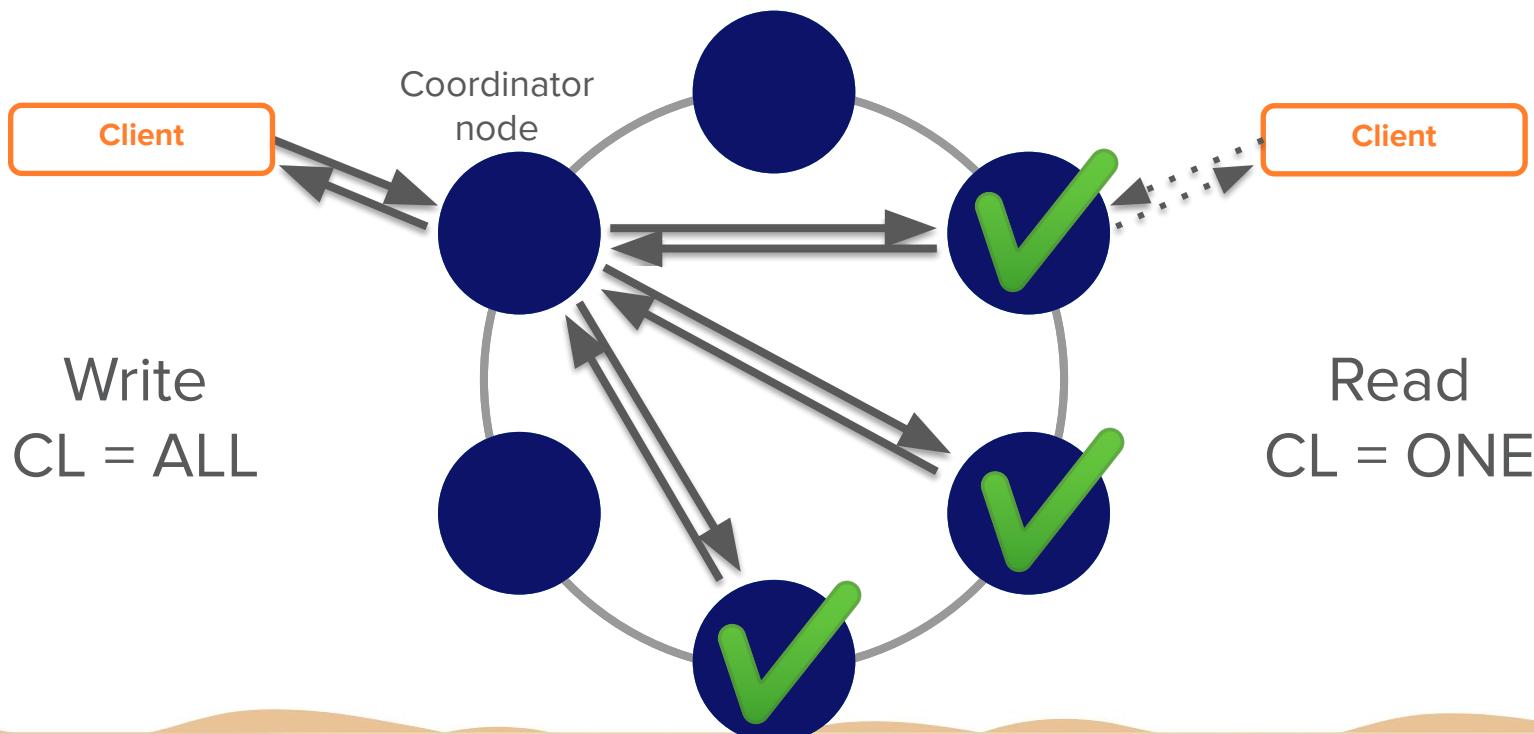
# Consistency Levels



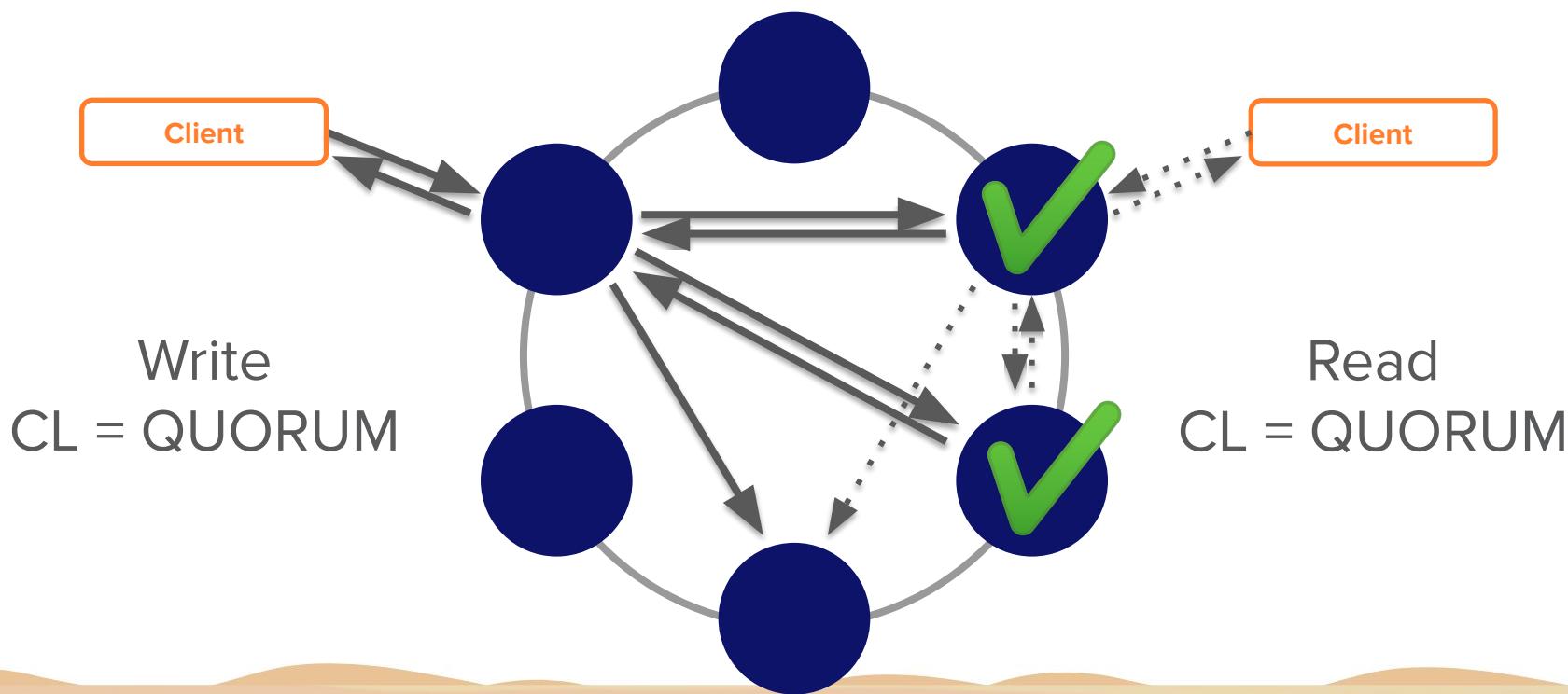
# Immediate Consistency


$$CL_{read} + CL_{write} > RF$$

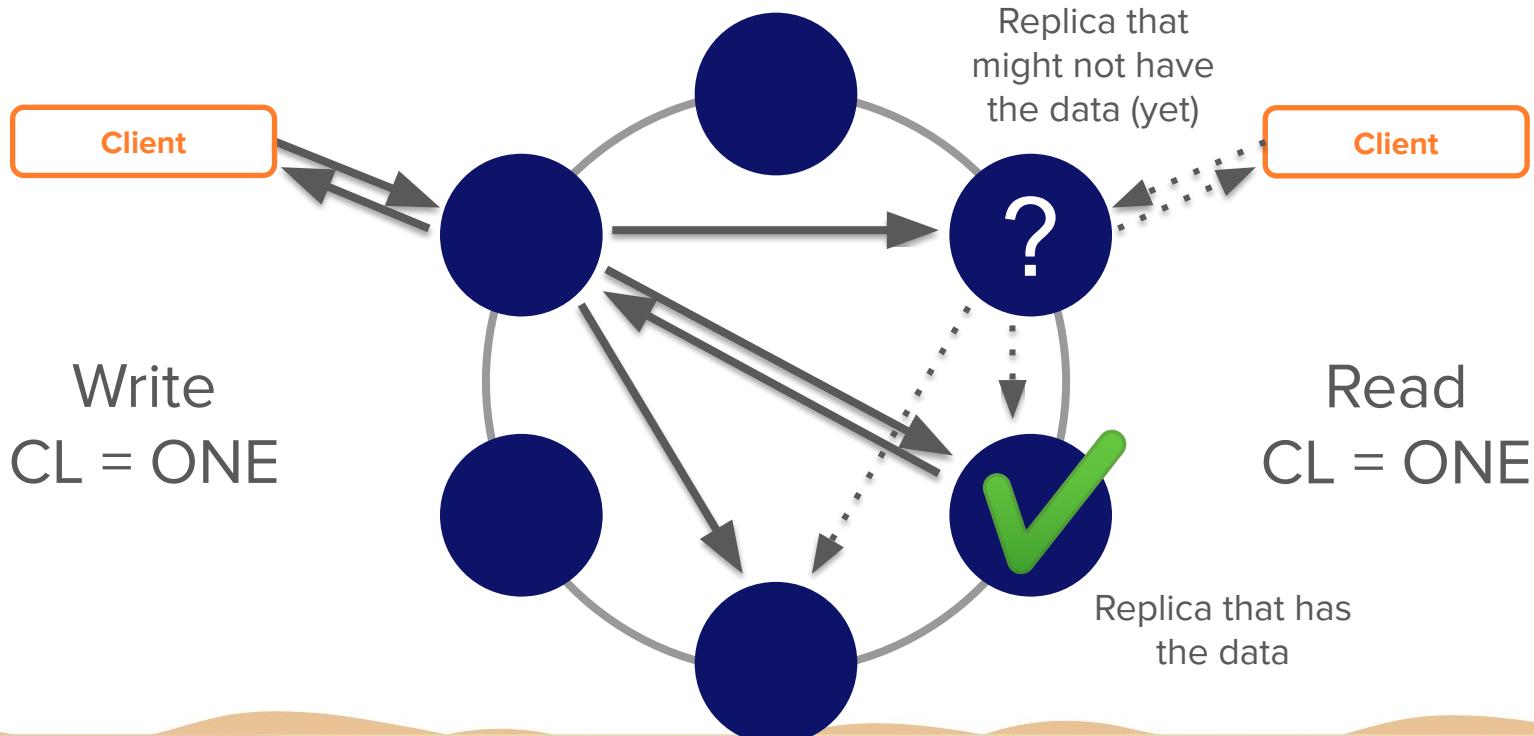
# Immediate Consistency – One Way



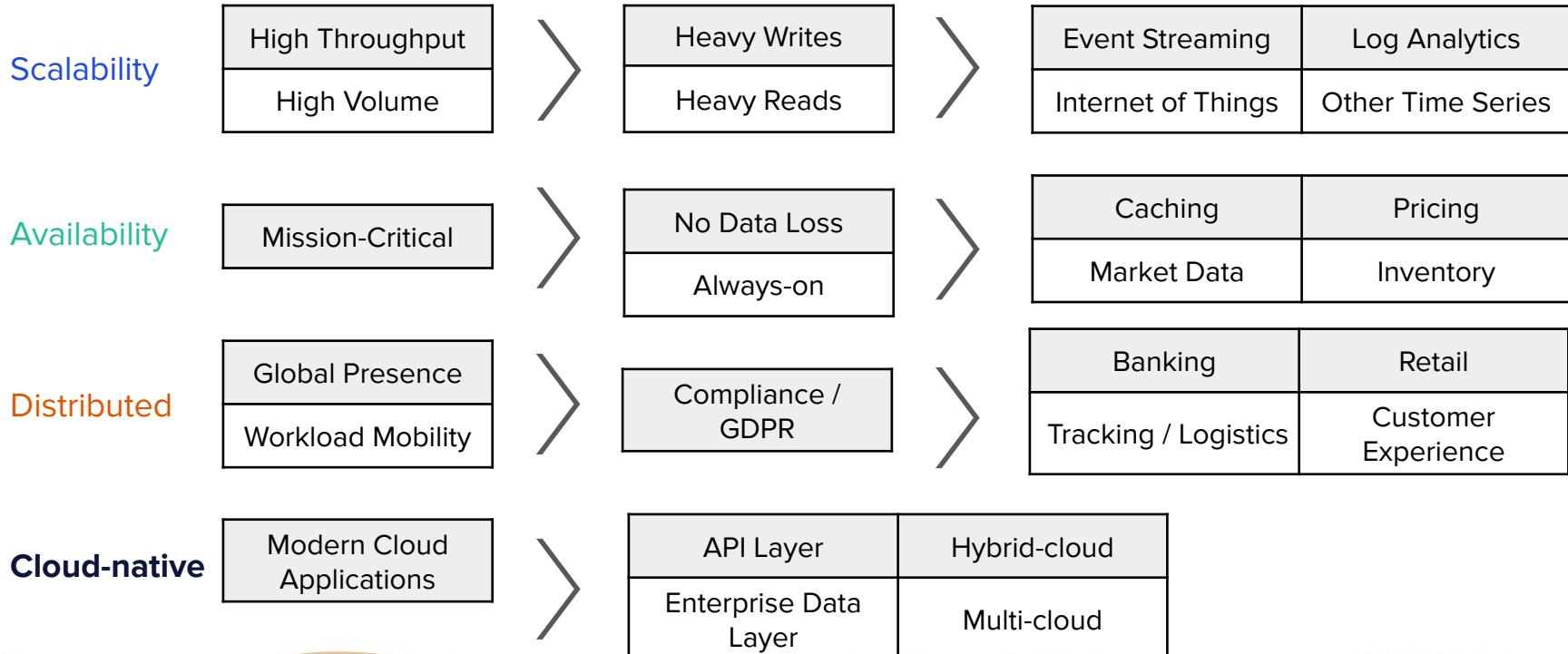
# Immediate Consistency – A Better Way



# Weak(er) Consistency



# Understanding Use Cases



# Exercise

DataStax Studio



The screenshot shows the DataStax Studio interface with the title "Cassandra Developer Workshop #2 - Datastax Studio". The main content area is titled "Use the Markdown Editor". It contains instructions for the user:

In this section, you will do the following things:

- Expand the cell to see the markdown code editor
- Edit the markdown
- Render the markdown to show your changes

To start with, there are two sections to cells, the code editor and the results section. You are currently looking at the results section and the code editor is hidden.

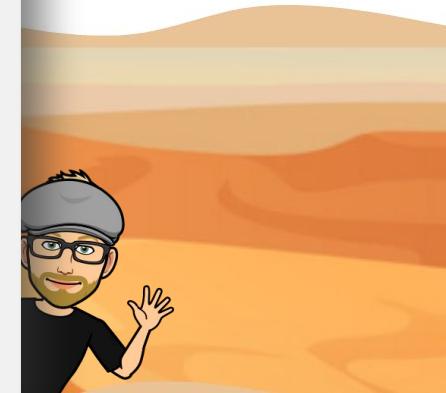
**Step 1:** Let's switch to the code editor. Hover over the right-hand corner of this cell and click the icon that looks like an eye.

A callout bubble points to the eye icon in the top right corner of the code editor window. The window itself has a "Language: Markdown" dropdown and a text input field.

Now, let's make a change to the markdown to see how it works.

**Step 2:** Scroll the bottom of the code editor window and find "Hello your\_name\_goes\_here". Replace `your_name_goes_here` with your name.

A callout bubble points to a button labeled "Replace this text with your name".





# Developer Workshop Series

- Bootstrapping
- Apache Cassandra™ Why, What & When
- Read and Write path
- Uber High Level Data Modeling
- What's NEXT?

# Write path



# Write Path

RAM

DISK

# Write Path

RAM

1 Dev Awesome TX Houston



DISK

# Write Path

MemTable

RAM

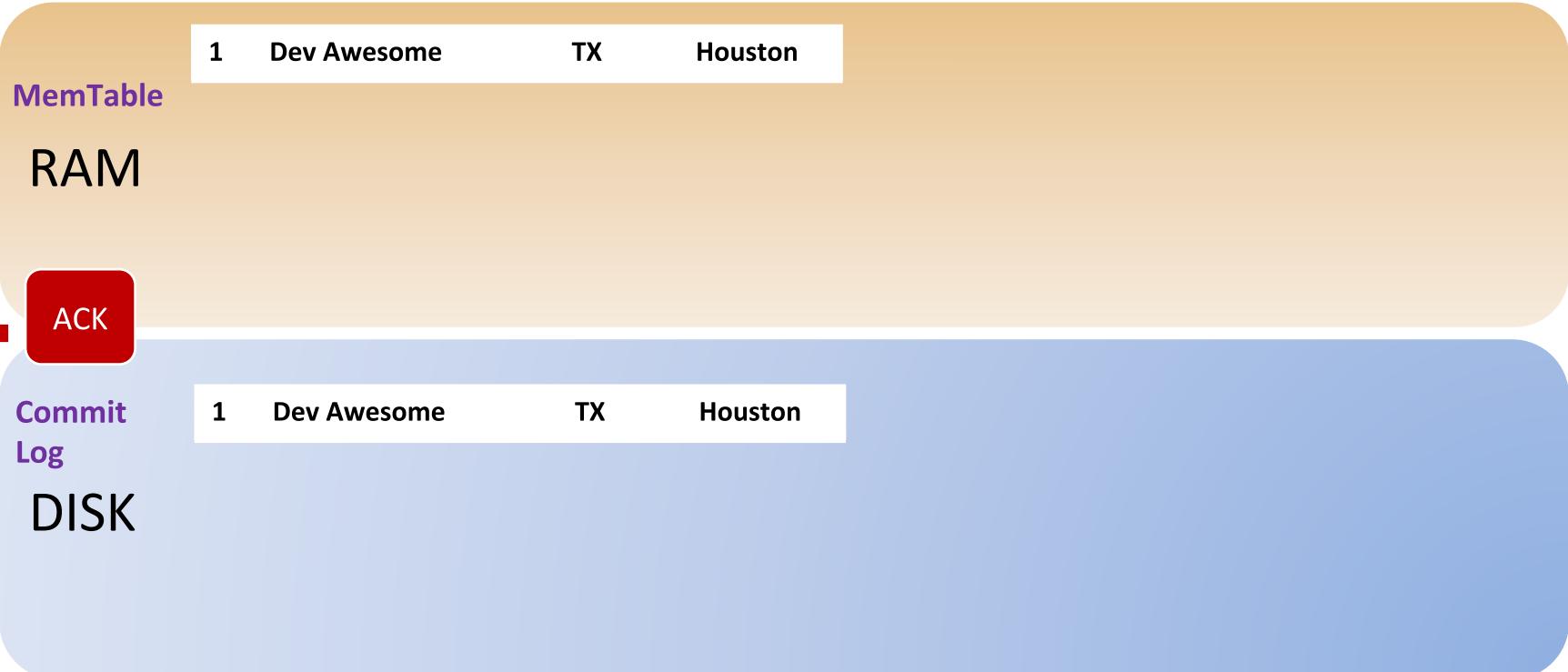
1	Dev Awesome	TX	Houston
---	-------------	----	---------

Commit Log

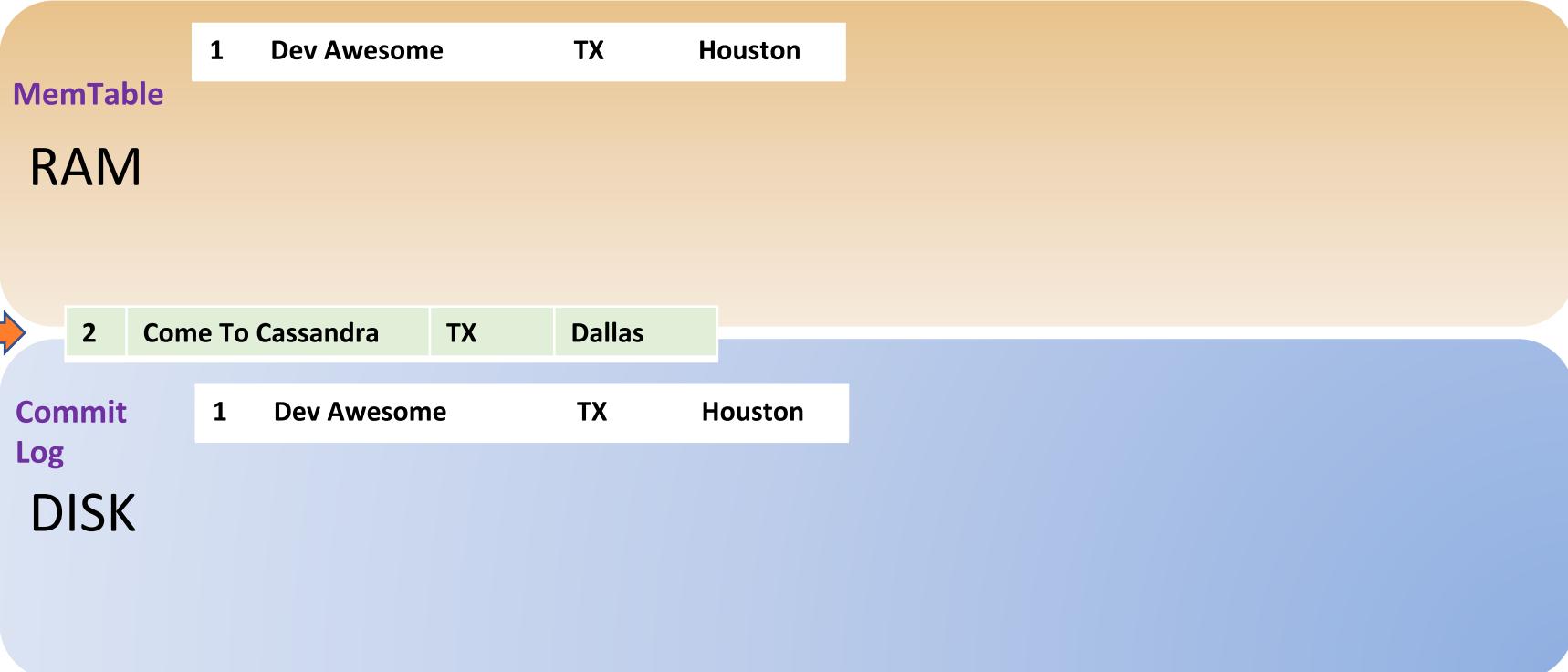
DISK

1	Dev Awesome	TX	Houston
---	-------------	----	---------

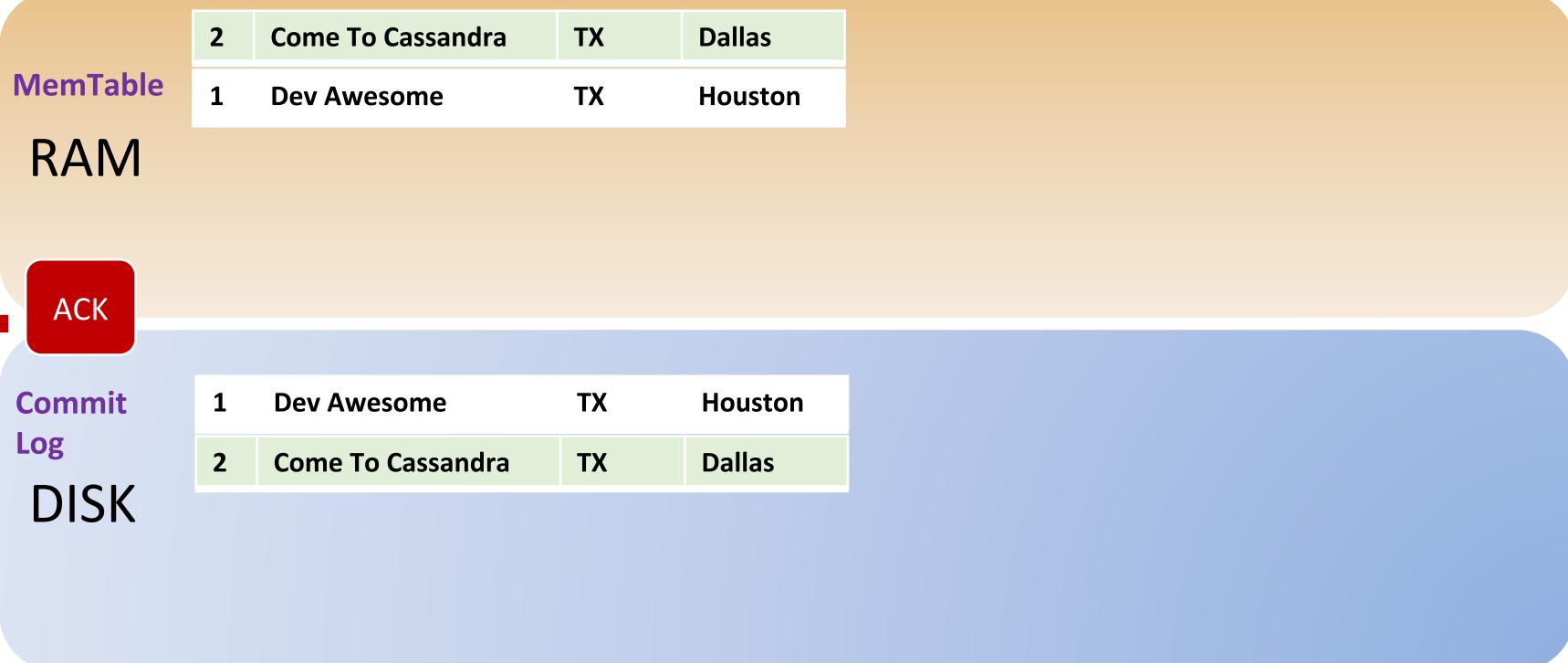
# Write Path



# Write Path



# Write Path



# Write Path

MemTable

2	Come To Cassandra	TX	Dallas
1	Dev Awesome	TX	Houston

RAM

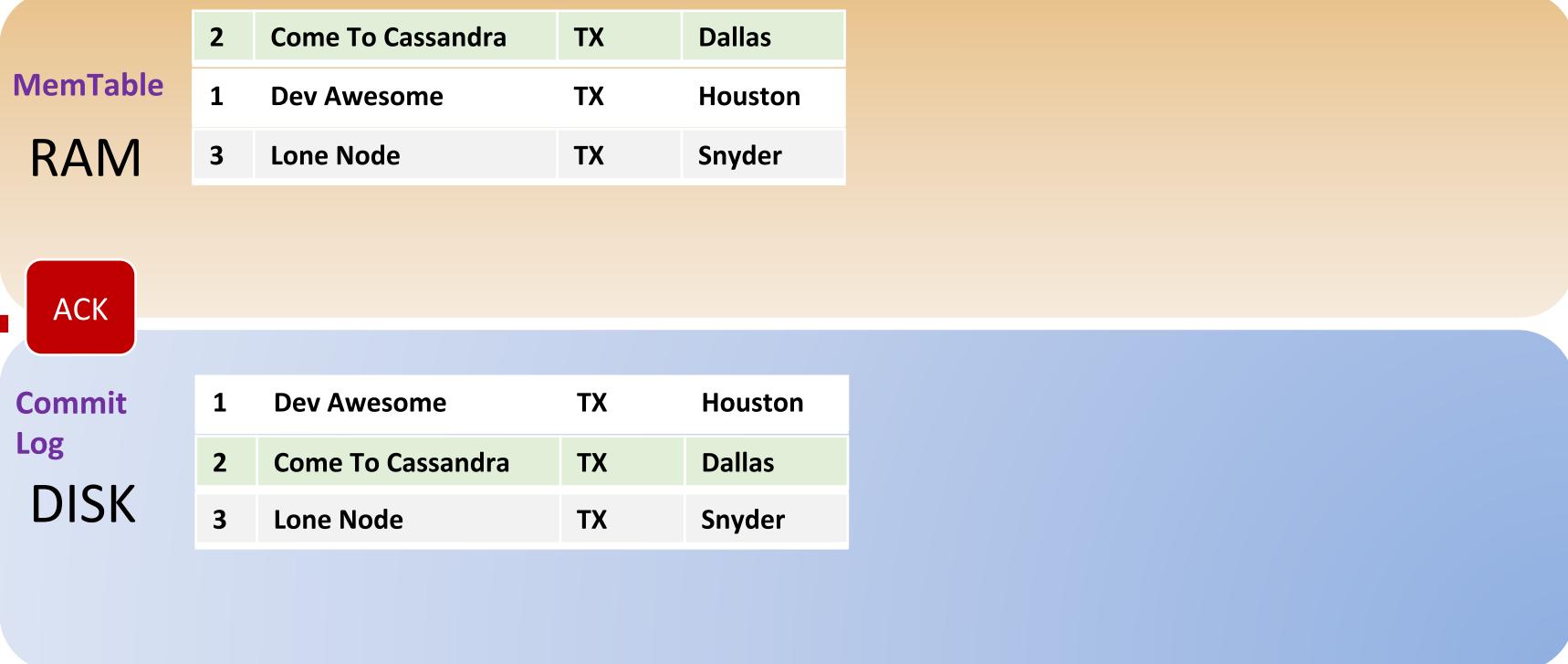


Commit  
Log

3	Lone Node	TX	Snyder
1	Dev Awesome	TX	Houston
2	Come To Cassandra	TX	Dallas

DISK

# Write Path



# Write Path

MemTable RAM	2	Come To Cassandra	TX	Dallas
	1	Dev Awesome	TX	Houston
	3	Lone Node	TX	Snyder

→

Commit Log DISK	4	IgotUr Data	TX	Austin
	1	Dev Awesome	TX	Houston
	2	Come To Cassandra	TX	Dallas
	3	Lone Node	TX	Snyder

# Write Path

MemTable  
RAM

4	IgotUr Data	TX	Austin
2	Come To Cassandra	TX	Dallas
1	Dev Awesome	TX	Houston
3	Lone Node	TX	Snyder

ACK

Commit  
Log  
DISK

1	Dev Awesome	TX	Houston
2	Come To Cassandra	TX	Dallas
3	Lone Node	TX	Snyder
4	IgotUr Data	TX	Austin

# Write Path

MemTable

RAM

4	IgotUr Data	TX	Austin
2	Come To Cassandra	TX	Dallas
1	Dev Awesome	TX	Houston
3	Lone Node	TX	Snyder

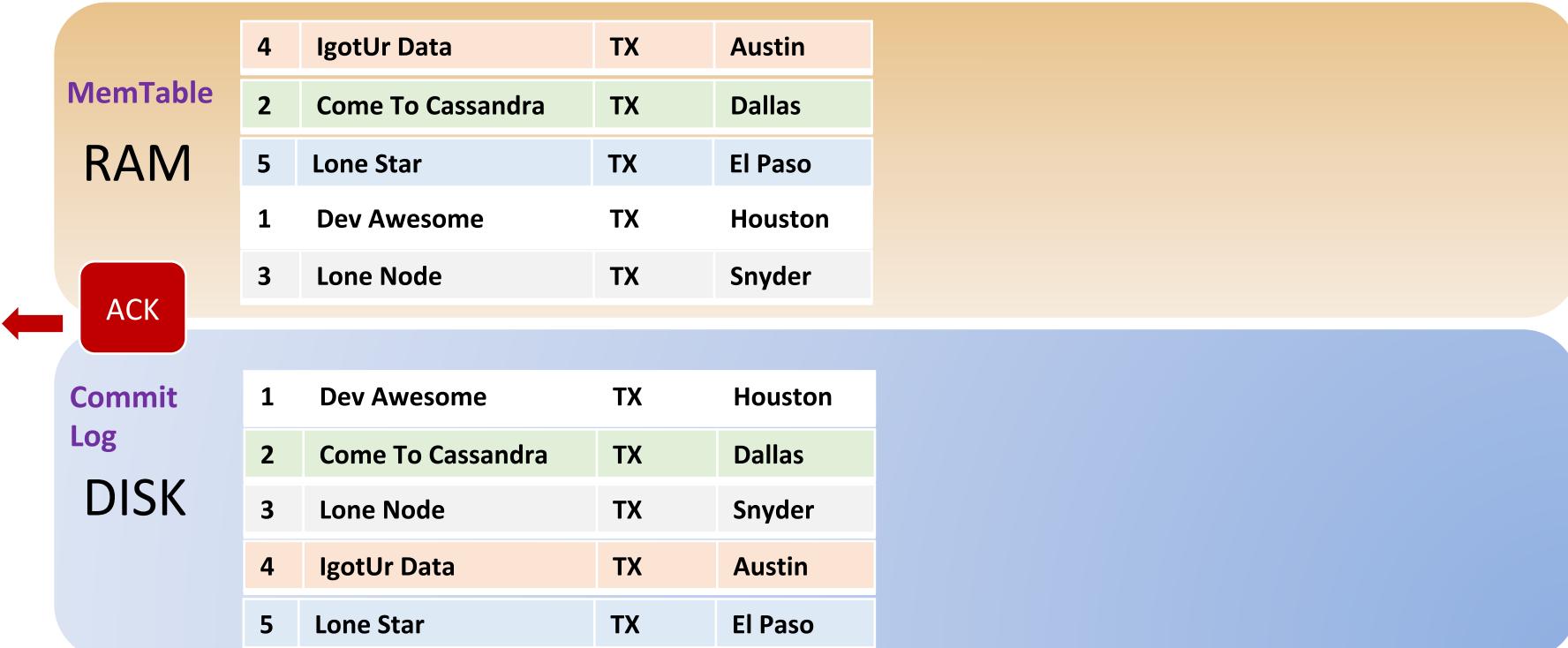
5 Lone Star TX El Paso

Commit Log

DISK

1	Dev Awesome	TX	Houston
2	Come To Cassandra	TX	Dallas
3	Lone Node	TX	Snyder
4	IgotUr Data	TX	Austin

# Write Path



# Write Path

MemTable  
RAM

4	IgotUr Data	TX	Austin
2	Come To Cassandra	TX	Dallas
5	Lone Star	TX	El Paso
1	Dev Awesome	TX	Houston
3	Lone Node	TX	Snyder

FLUSH

Commit  
Log  
DISK

1	Dev Awesome	TX	Houston
2	Come To Cassandra	TX	Dallas
3	Lone Node	TX	Snyder
4	IgotUr Data	TX	Austin
5	Lone Star	TX	El Paso

SSTABLE

4	IgotUr Data	TX	Austin
2	Come To Cassandra	TX	Dallas
5	Lone Star	TX	El Paso
1	Dev Awesome	TX	Houston
3	Lone Node	TX	Snyder

# Write Path

MemTable

RAM

Commit Log

DISK

1	Dev Awesome	TX	Houston
2	Come To Cassandra		Dallas
3	Lone Node	X	Snyder
4	IgotUr Data		Austin
5	Lone Star	TX	El Paso



4	IgotUr Data	TX	Austin
2	Come To Cassandra	TX	Dallas
5	Lone Star	TX	El Paso
1	Dev Awesome	TX	Houston
3	Lone Node	TX	Snyder

# Write Path

MemTable

RAM

DISK

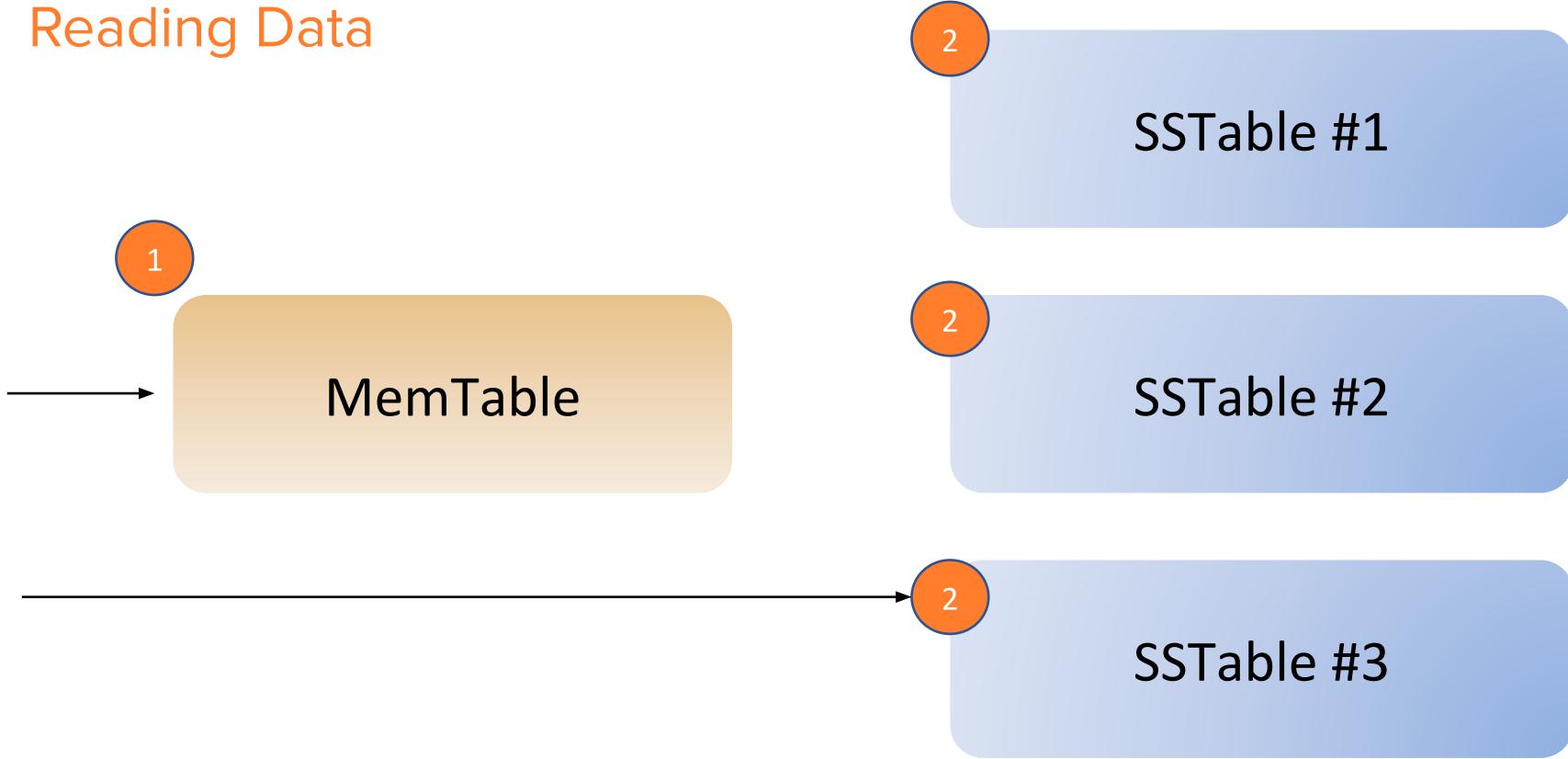
SSTABLE  
(IMMUTABLE)

4	IgotUr Data	TX	Austin
2	Come To Cassandra	TX	Dallas
5	Lone Star	TX	El Paso
1	Dev Awesome	TX	Houston
3	Lone Node	TX	Snyder

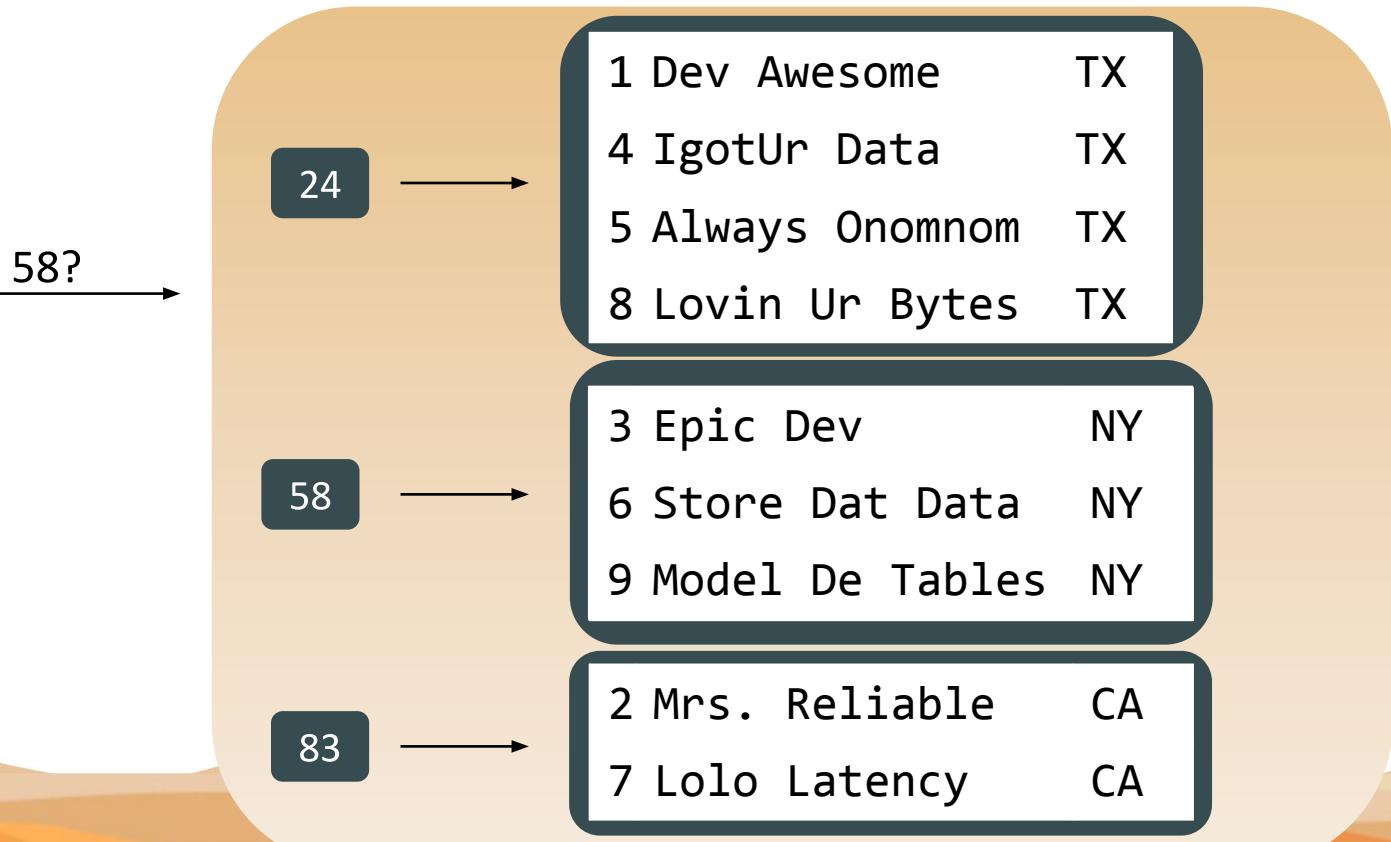
# Read path



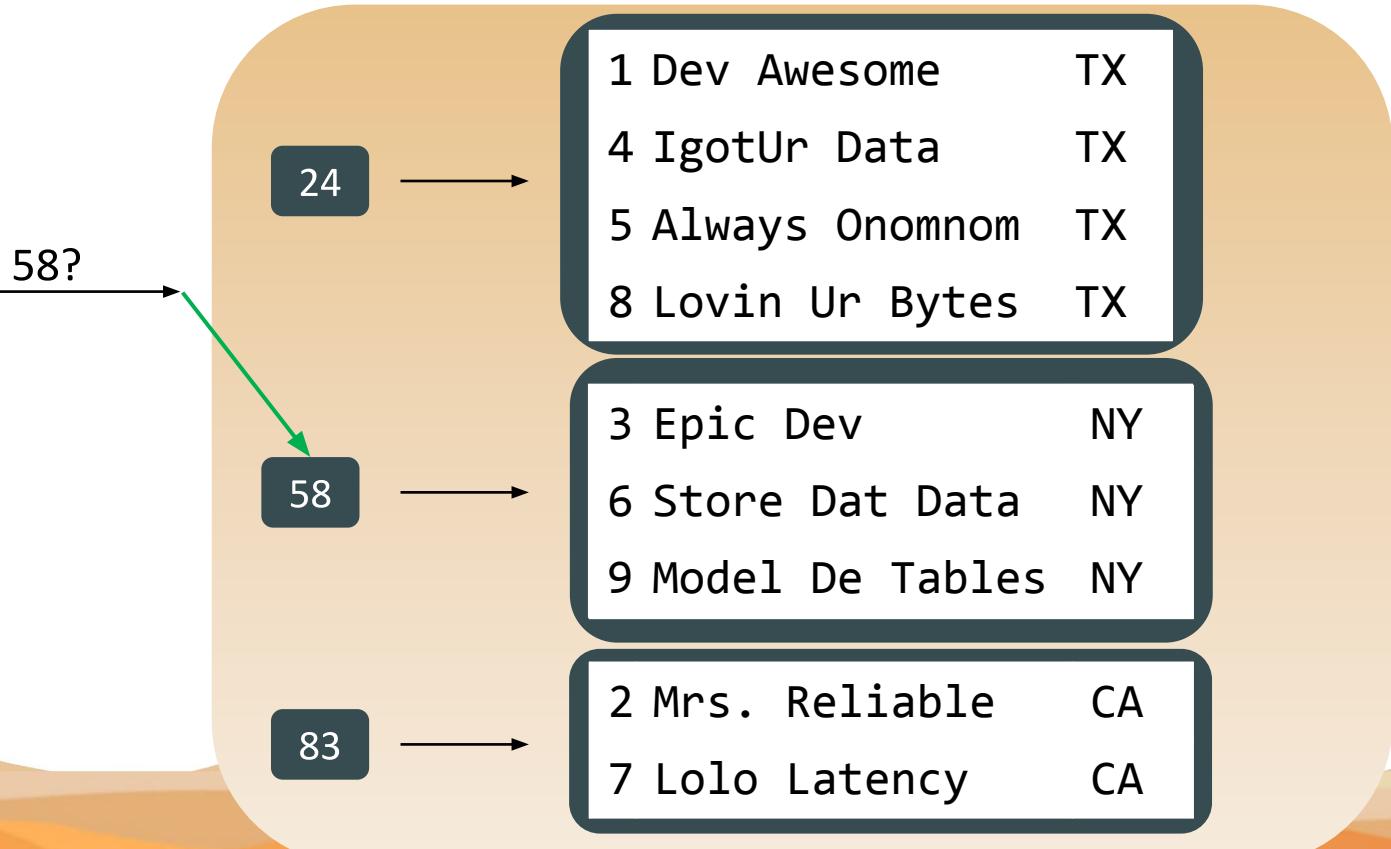
# Reading Data



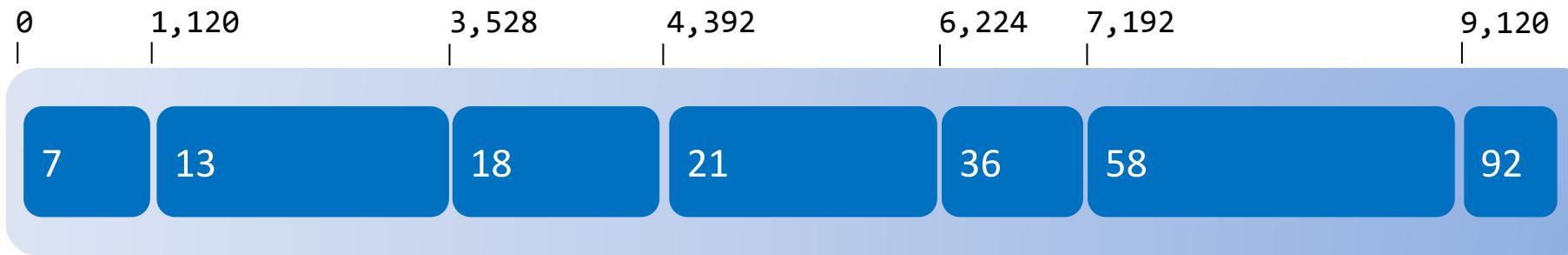
# Reading a MemTable



# Reading a MemTable

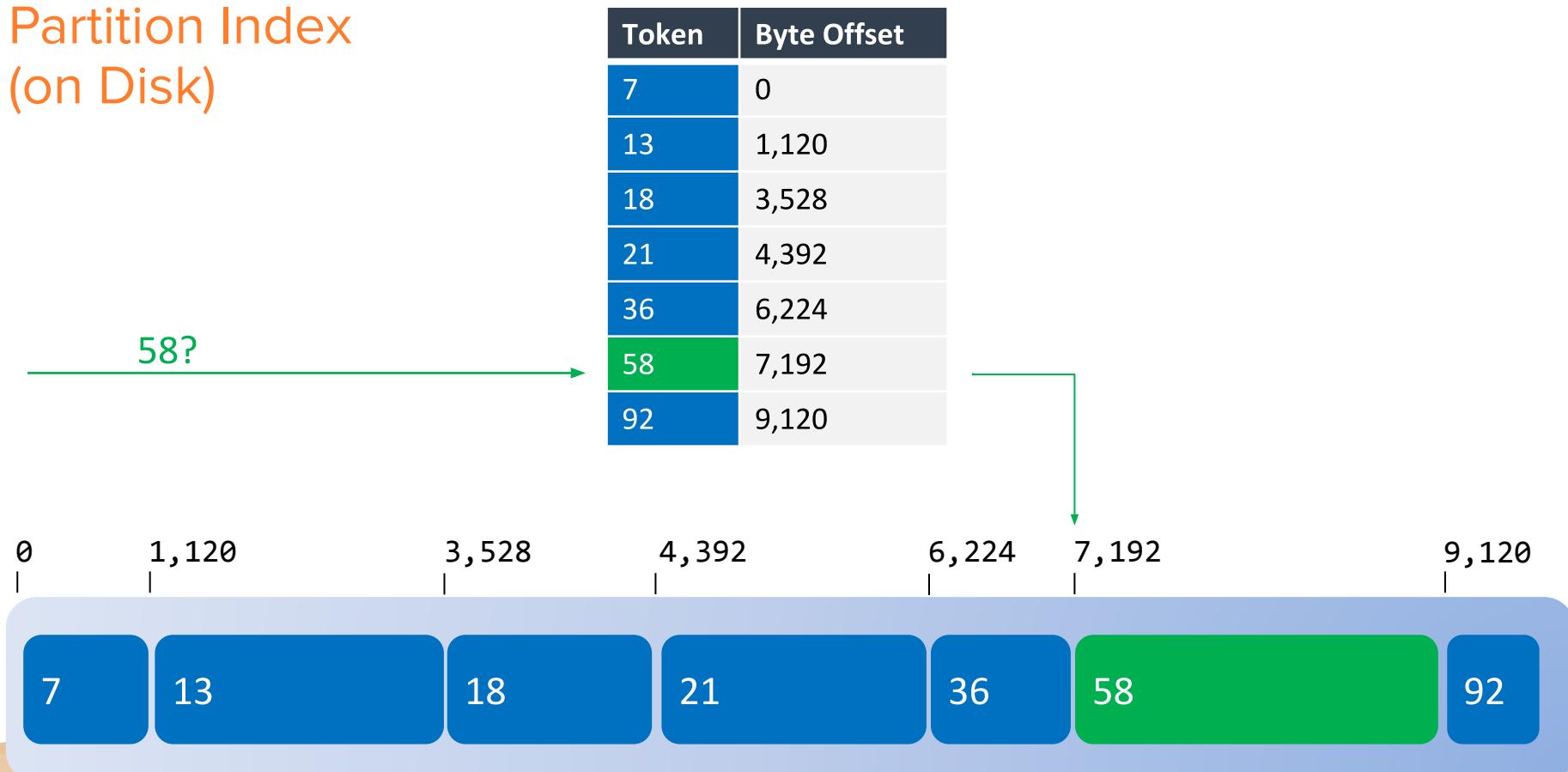


# Reading a SSTable

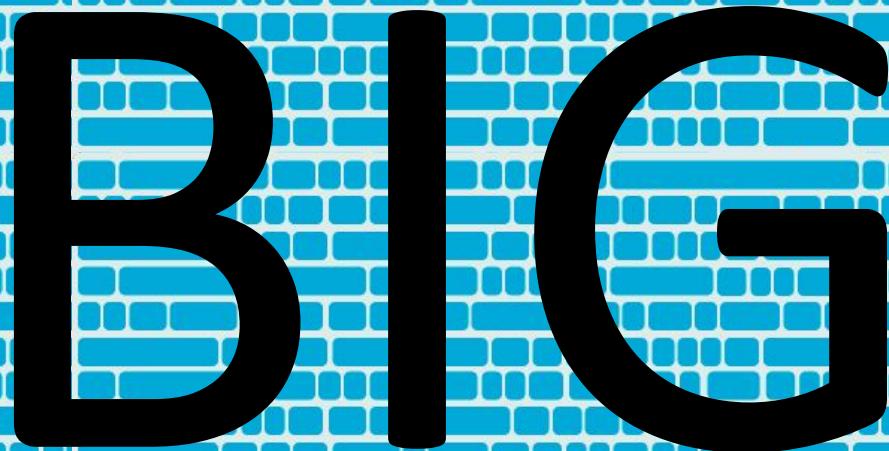


- ❖ A SSTable holds ordered partitions
- ❖ A partition can be split in multiple SSTables
- ❖ We can mark offset of each partition

# Partition Index (on Disk)

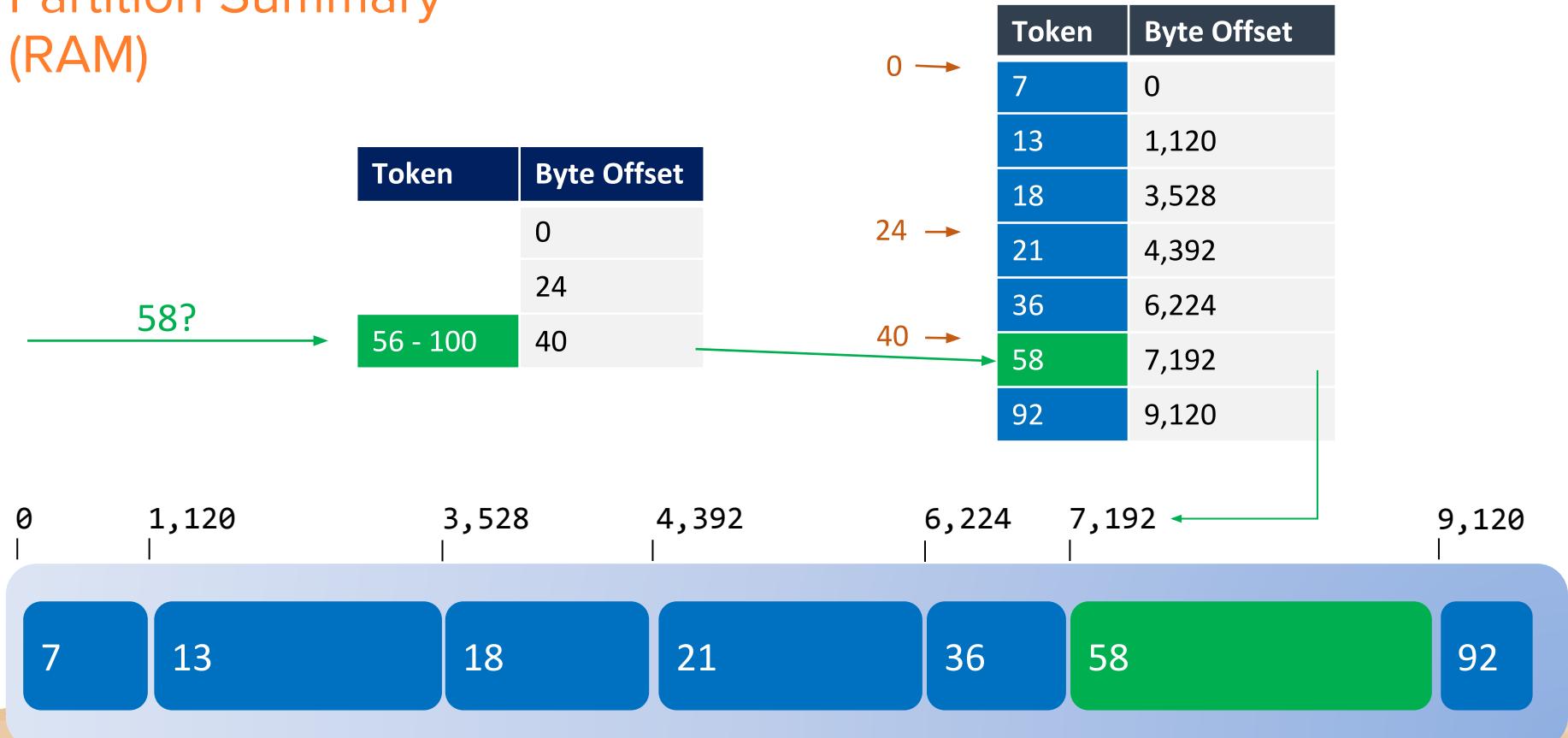


...but a SSTABLE is ...



**BIG**

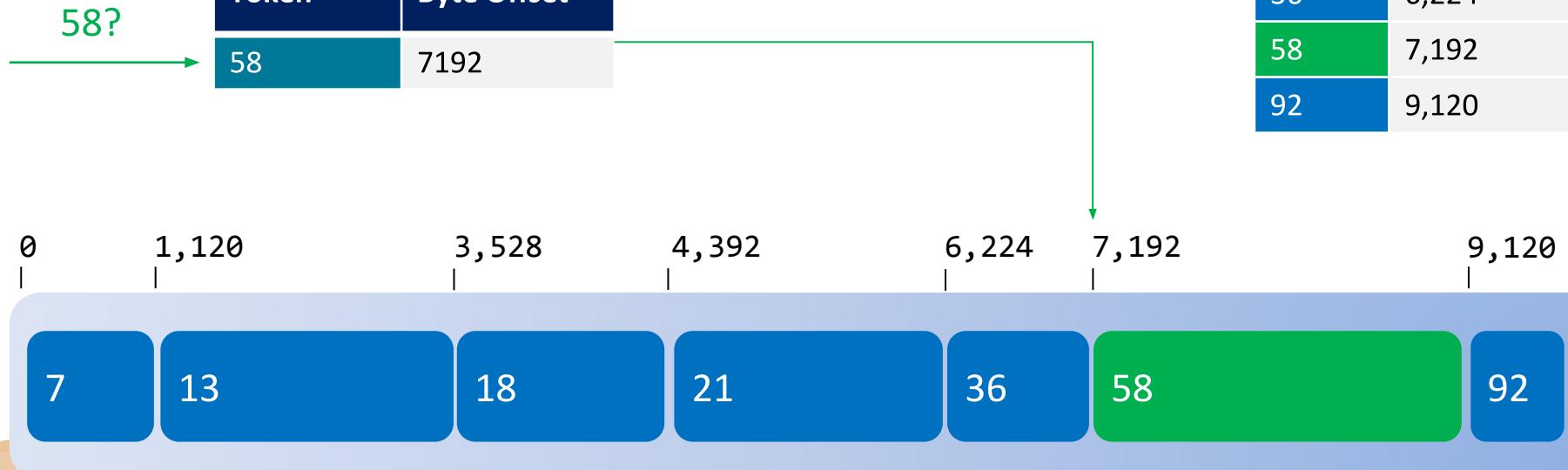
# Partition Summary (RAM)



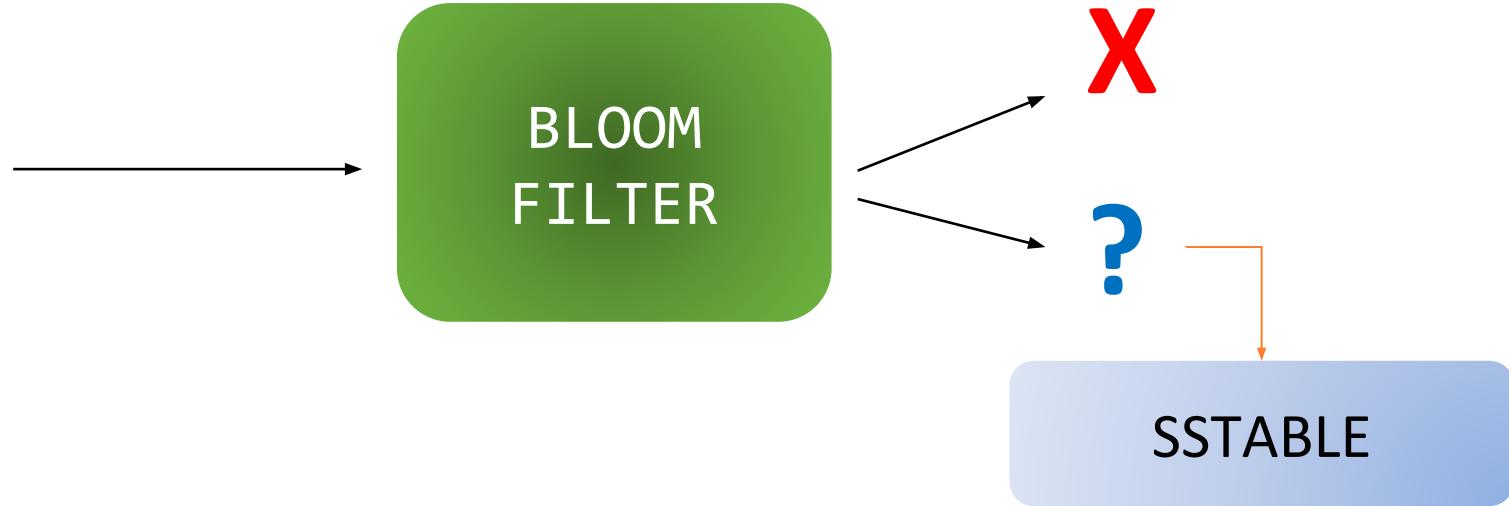
# Key Cache (RAM)

Token	Byte Offset
0	
24	
40	

Token	Byte Offset
58	7192



# Bloom Filter

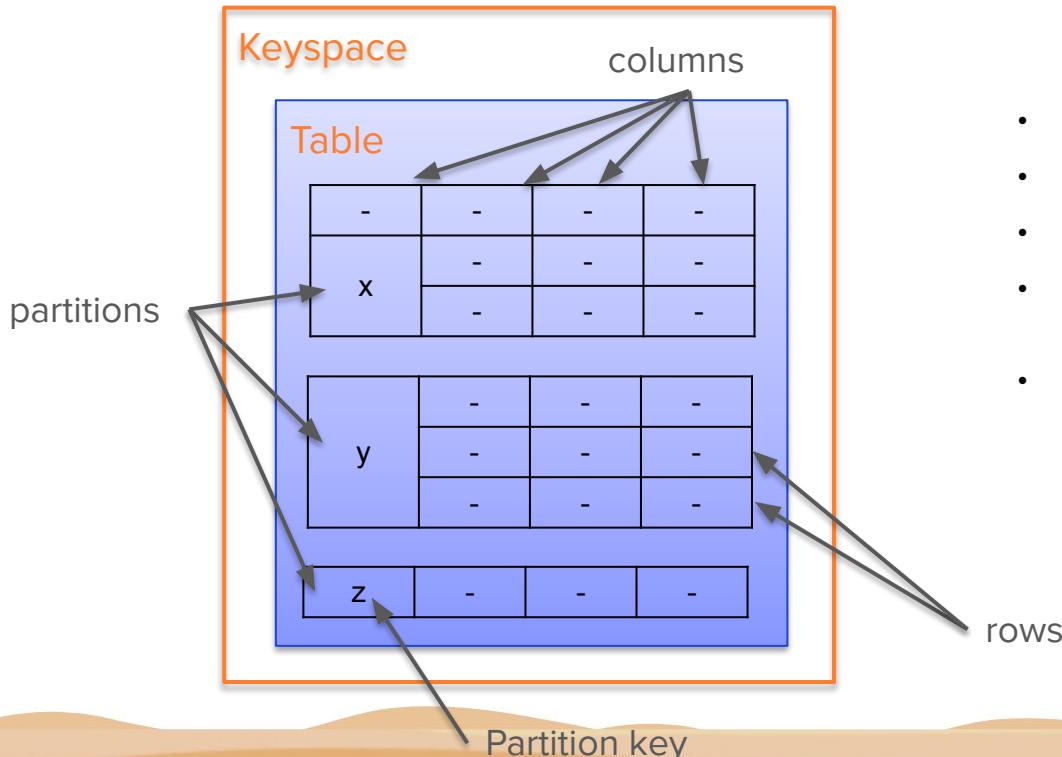




# Developer Workshop Series

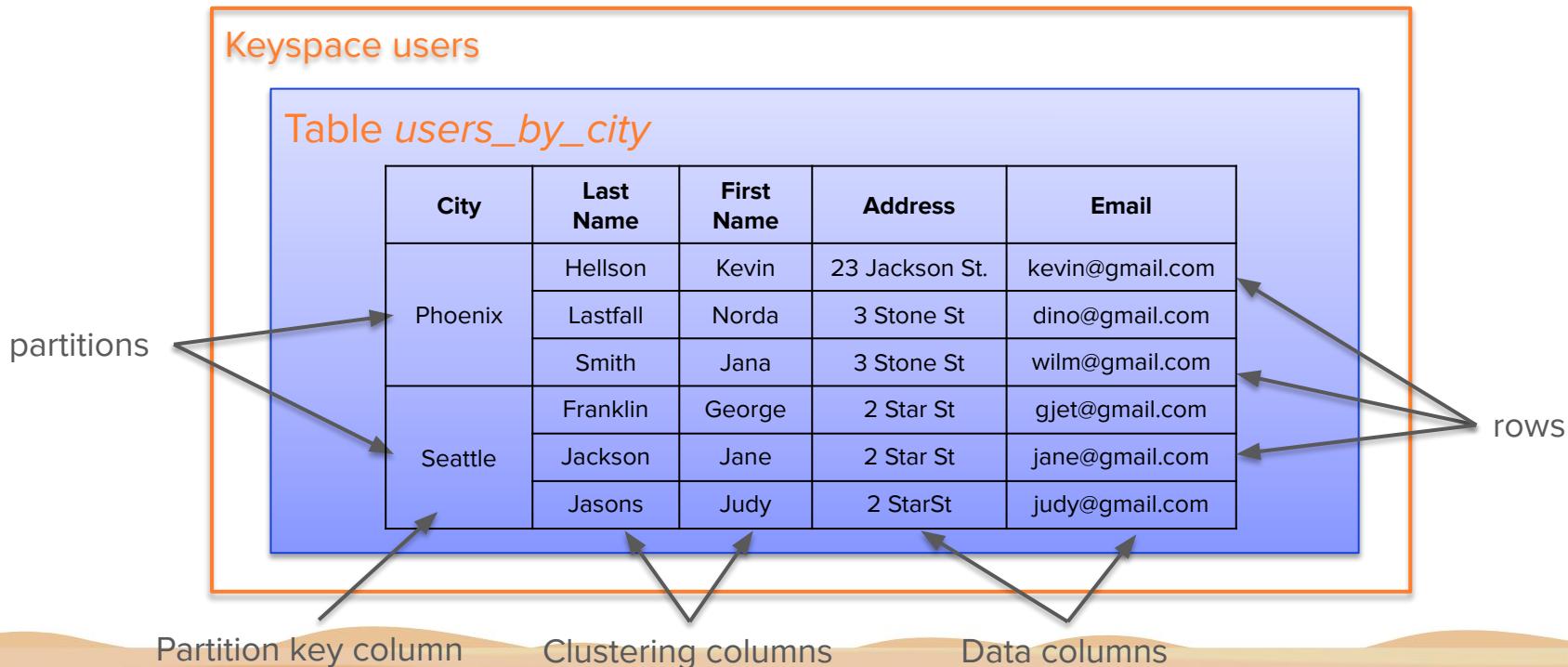
- Bootstrapping
- Apache Cassandra™ Why, What & When
- Read and Write path
- Uber High Level Data Modeling
- What's NEXT?

# Cassandra Structure - Partition



- Tabular data model, with one twist
- *Keyspaces* contain *tables*
- *Tables* are organized in *rows* and *columns*
- Groups of related rows called *partitions* are stored together on the same node (or nodes)
- Each row contains a *partition key*
  - One or more columns that are hashed to determine which node(s) store that data

# Example Data – Users organized by city



# Tables Hold Many Partitions

City	Last Name	First Name	Address	Email
Phoenix	Hellson	Kevin	23 Jackson St.	kevin@gmail.com
	Lastfall	Norda	3 Stone St	dino@gmail.com
	Smith	Jana	3 Stone St	wilm@gmail.com

Table *users\_by\_city*

# Tables Hold Many Partitions

City	Last Name	First Name	Address	Email
Seattle	Franklin	George	2 Star St	gjet@gmail.com
	Jackson	Jane	2 Star St	jane@gmail.com
	Jasons	Judy	2 StarSt	judy@gmail.com

Table *users\_by\_city*

City	Last Name	First Name	Address	Email
Phoenix	---	---	---	---
	---	---	---	---
	---	---	---	---

# Tables Hold Many Partitions

City	Last Name	First Name	Address	Email
Charlotte	Azrael	Chris	5 Blue St	chris@gmail.com
	Stilson	Brainy	7 Azure Ln	brain@gmail.com
	Smith	Cristina	4 Teal Cir	clu@gmail.com
	Sage	Grant	9 Royal St	grant@gmail.com
	Seterson	Peter	2 Navy Ct	peter@gmail.com

Table *users\_by\_city*

City	Last Name	First Name	Address	Email
Phoenix	---	---	---	---
	---	---	---	---
	---	---	---	---

City	Last Name	First Name	Address	Email
Seattle	---	---	---	---
	---	---	---	---
	---	---	---	---

# Tables Hold Many Partitions

Table *users\_by\_city*

City	Last Name	First Name	Address	Email
Phoenix	---	---	---	---
	---	---	---	---
	---	---	---	---

Seattle	---	---	---	---
	---	---	---	---
	---	---	---	---

Charlotte	---	---	---	---
	---	---	---	---
	---	---	---	---
	---	---	---	---
	---	---	---	---

# Creating a Keyspace in CQL

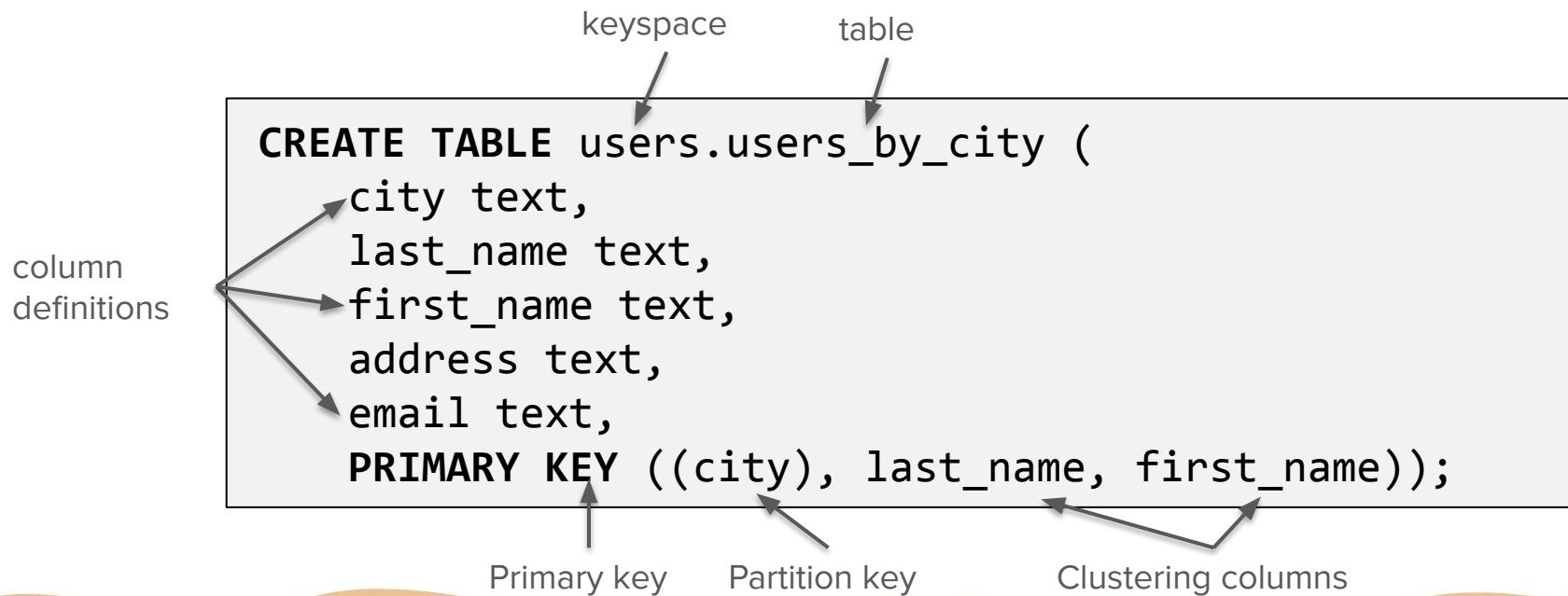
```
CREATE KEYSPACE users
  WITH REPLICATION = {
    'class' : 'NetworkTopologyStrategy',
    'datacenter1' : 3
};
```

keyspace

replication strategy

Replication factor by data center

# Creating a Table in CQL



# Exercise

## Working with CQL



The screenshot shows the DataStax Studio interface for a Cassandra Developer Workshop. The title bar reads "Cassandra Developer Workshop #3 - Working with CQL". The main area is a code editor with the following content:

```
Language: Markdown
DESCRIBE KEYSPACE Command

In this section, you will do the following things:
• Inspect the killrvideo keyspace

Step 1: In the following CQL cell, execute a DESCRIBE KEYSPACE command for the killrvideo keyspace.

▶ Need a command hint? Click here.
▶ Just want to copy the command? Click here.
```

At the bottom of the code editor, it says "Language: CQL" and "Keyspace: killrvideo".

To the right of the code editor is a sidebar titled "Schema killrvideo". It shows a tree structure of the "killrvideo" keyspace, which contains the following tables:

- comments\_by\_user
- comments\_by\_video
- latest\_videos
- tags\_by\_letter
- user\_credentials
- user\_videos
- users
- video\_playback\_stats
- video\_ratings
- video\_ratings\_by\_user
- video\_recommendations
- video\_recommendations\_by\_video
- videos
- videos\_by\_tag

Below the tables, there are sections for User Defined Types, Functions, Aggregates, and Materialized Views.



# Developer Workshop Series

**What we will  
cover:**

- Bootstrapping
- Apache Cassandra™ Why, What & When
- Read and Write path
- Uber High Level Data Modeling
- What's NEXT?



# Homework Week 1



1. Finish notebooks if you are not done
2. Please provide feedback to ASTRA team with the FORM
3. Create Accounts on Academy and Community
4. Watch DS201 (videos | quizz)
5. Complete exercise of the week (3 questions)



# Developer Resources

**LEARN**

Join [academy.datastax.com](https://academy.datastax.com)

<https://academy.datastax.com/resources/cassandra-developer-workshop>

Free online courses - Cassandra certifications

**ASK/SHARE**

Join [community.datastax.com](https://community.datastax.com)

Ask/answer community user questions - share your expertise

**CONNECT**

Follow us

We are on Twitter - Twitch!

**REVIEW**

Slides and code for this course are available at

<https://github.com/DataStax-Academy/online-Cassandra-workshop>

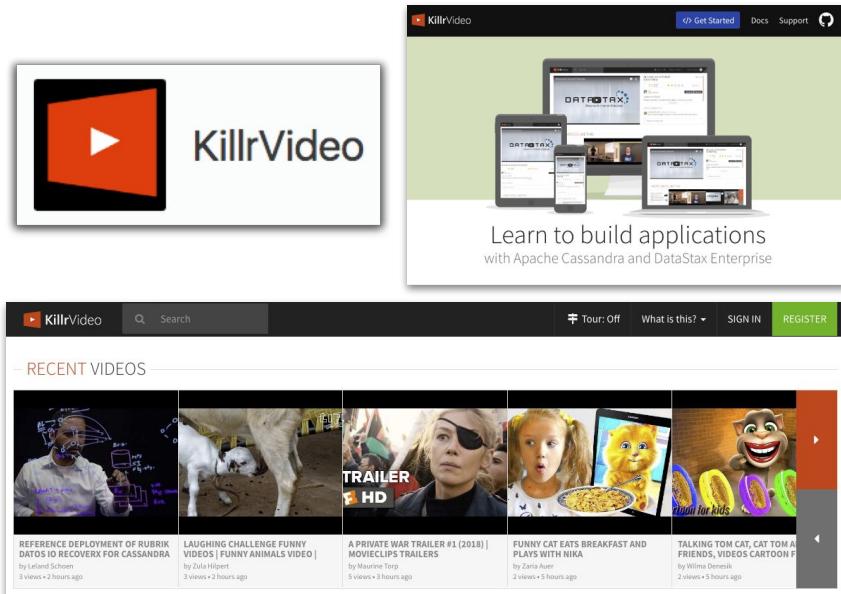
# Training Courses at DataStax Academy

- Free self-paced DSE 6 courses
  - ◆ [DS201: DataStax Enterprise 6 Foundations of Apache Cassandra™](#)



# KillrVideo Reference Application

- Reference application for learning how to use Apache Cassandra and DataStax Enterprise
  - ◆ DataStax Drivers
  - ◆ Docker images
- Source code freely available
  - ◆ <https://github.com/killrvideo>
- Live version
  - ◆ <http://killrvideo.com>
- Download, test, modify, contribute!



# Part 1 - How to build Applications with Cassandra

Every Wednesday (NAM/LATAM) - 9am PDT / 12pm EDT / 5:00pm BST / 6:00pm CEST / 21:30pm IST

July 1

Getting Started with Cassandra



July 8

Data Modelling for Apache Cassandra™

July 15

Application Development with Cassandra part 1

July 22

Application Development with Cassandra part 2

# Part 2 - Test, Deploy and Monitor

Every Wednesday (NAM/LATAM) - 9am PDT / 12pm EDT / 5:00pm BST / 6:00pm CEST / 21:30pm IST

July 29      Operating your Cassandra clusters

Aug 5      Running Cassandra performance tests

Aug 12      Testing your deployments and troubleshooting

Aug 19      Deploying Cassandra with Kubernetes

# Part 1 - How to build Applications with Cassandra

Every Thursday (EMEA/APAC) - 8am BST / 9:00am CEST / 12:30pm IST / 15:00pm SGT / 17:00 ACT

July 2

Getting Started with Cassandra



July 9

Data Modelling for Apache Cassandra™

July 16

Application Development with Cassandra part 1

July 23

Application Development with Cassandra part 2

# Part 2 - Test, Deploy and Monitor

Every Thursday (EMEA/APAC) - 8am BST / 9:00am CEST / 12:30pm IST / 15:00pm SGT / 17:00 ACT

July 30      Operating your Cassandra clusters

Aug 6      Running Cassandra performance tests

Aug 13      Testing your deployments and troubleshooting

Aug 20      Deploying Cassandra with Kubernetes



# Thank You

