

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



Jack Fryer

Community Manager



Aleksandr Volochnev

Developer Advocate





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?

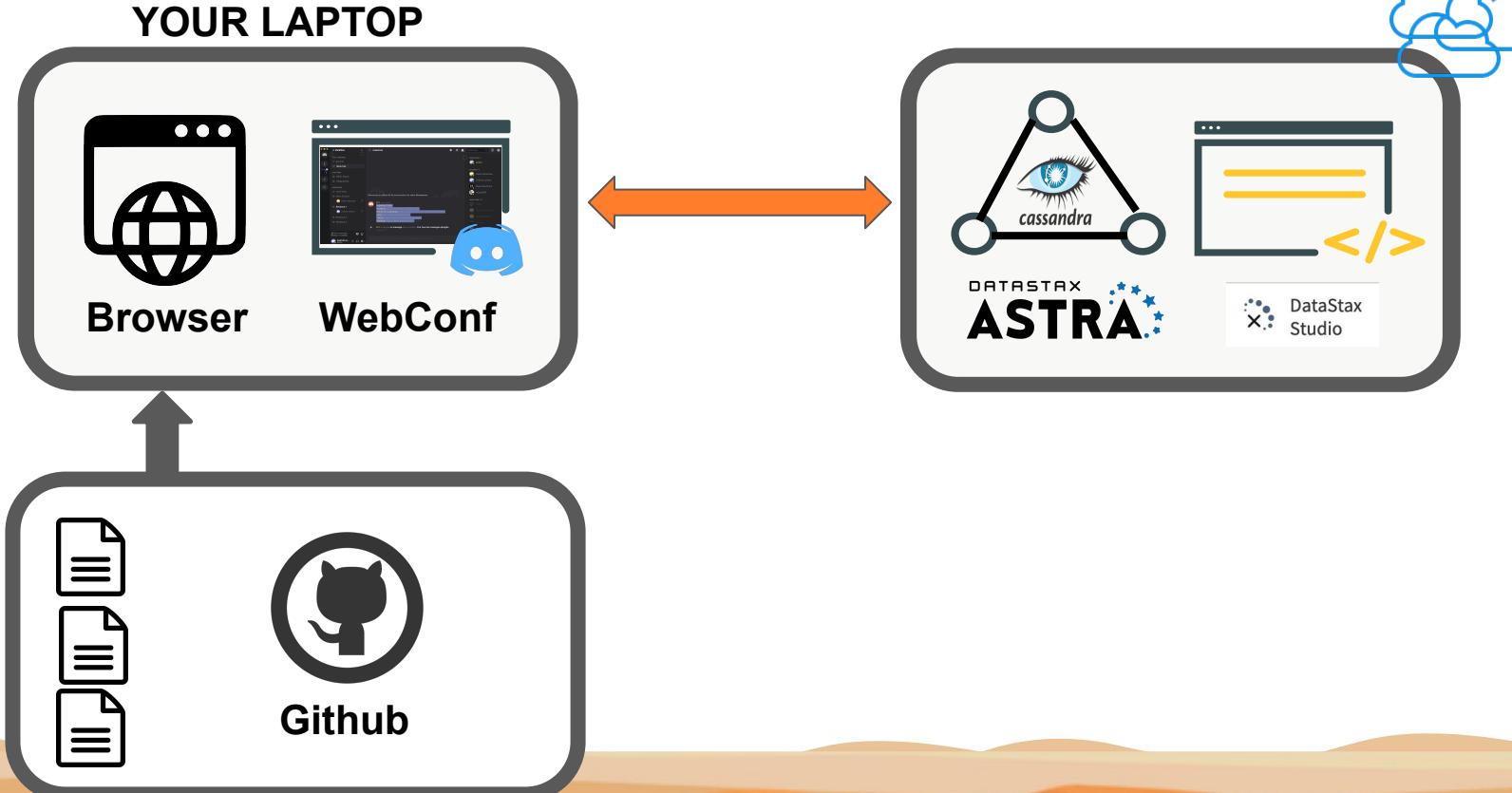


Developer Workshop Series

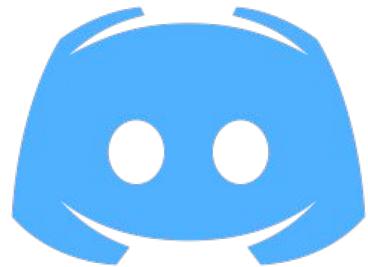
**What we will
cover:**

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

Overview



Get your hands on Discord



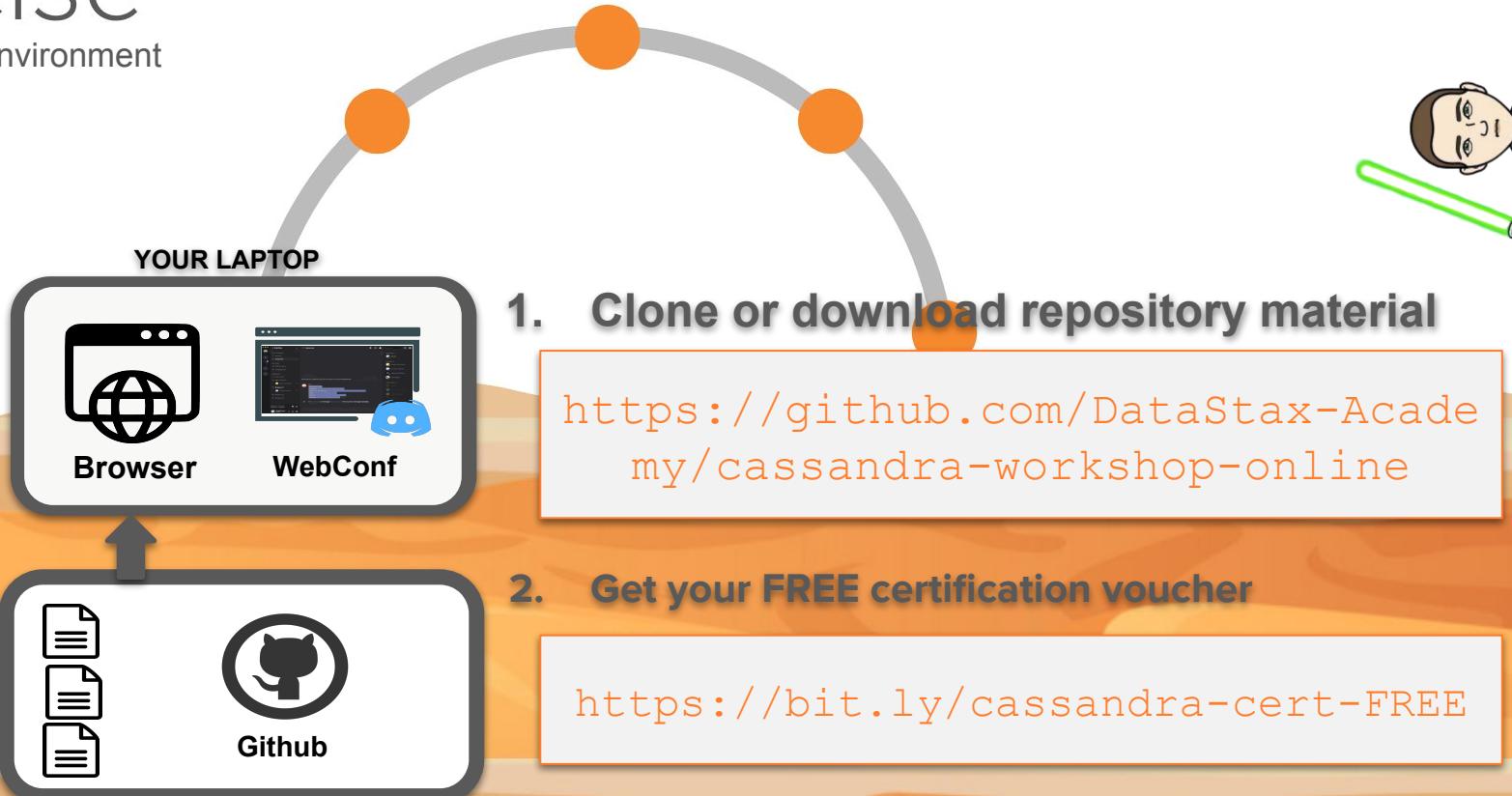
YOUR LAPTOP



- What you need to know
- List of resources available for you
- How to ask questions
- How to get the exercises
- How to mark the Exercises done

Exercise

Bootstrap your environment



Introducing Astra



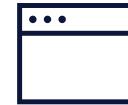
Eliminate Operations

everything from provisioning to backups is fully automated



Secure Your Data

with the most advanced security available for Cassandra



Simplify App Development

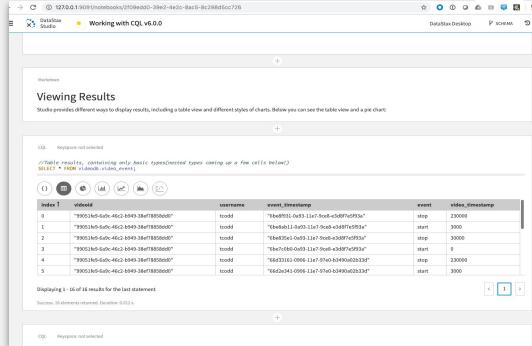
with auto-configured developer tools that deploy with a click

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



Great Drivers



Exercise

Creating your Astra instance



DATASTAX CONSTELLATION Your Organization

Provide Feedback cedrick.lunven@datastax.com ▾

2 Databases

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 Actions ▾

screenshot

Keyspace Name: okidoki
Organization: Your Organization

Owner: Cedrick Lunven
Created: January 22, 2020

Size and Location

Capacity Unit	500 GB Total Storage	Storage Used
1	500 GB Total Storage	

Locations: us-east-1 (1 capacity unit)
Compute Size: Startup
Replication Factor: 3

Cost

Spent this month: TBD

Estimated Cost

Per Hour Per Month

when running	TBD/hour
when parked	TBD/hour

Connection Details

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



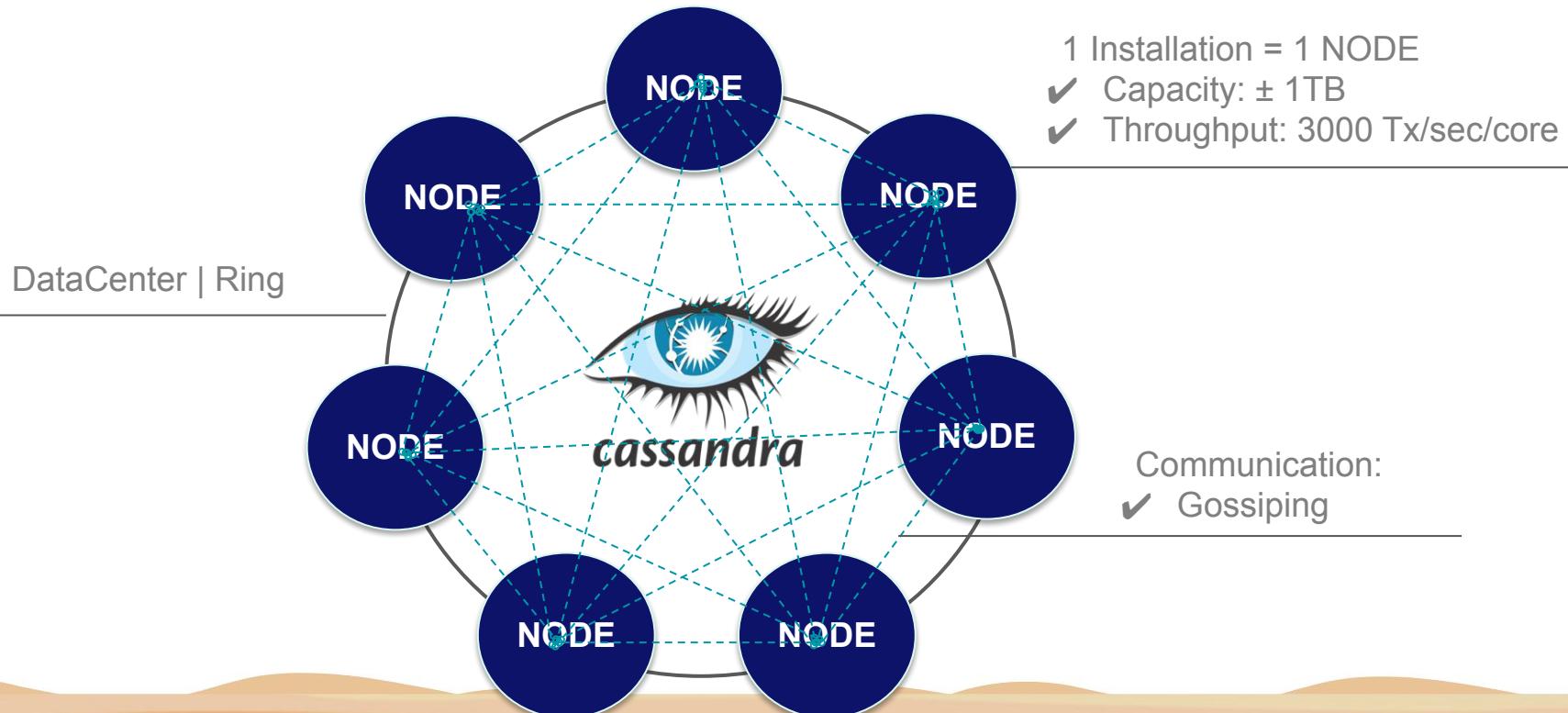


Developer Workshop Series

**What we will
cover:**

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

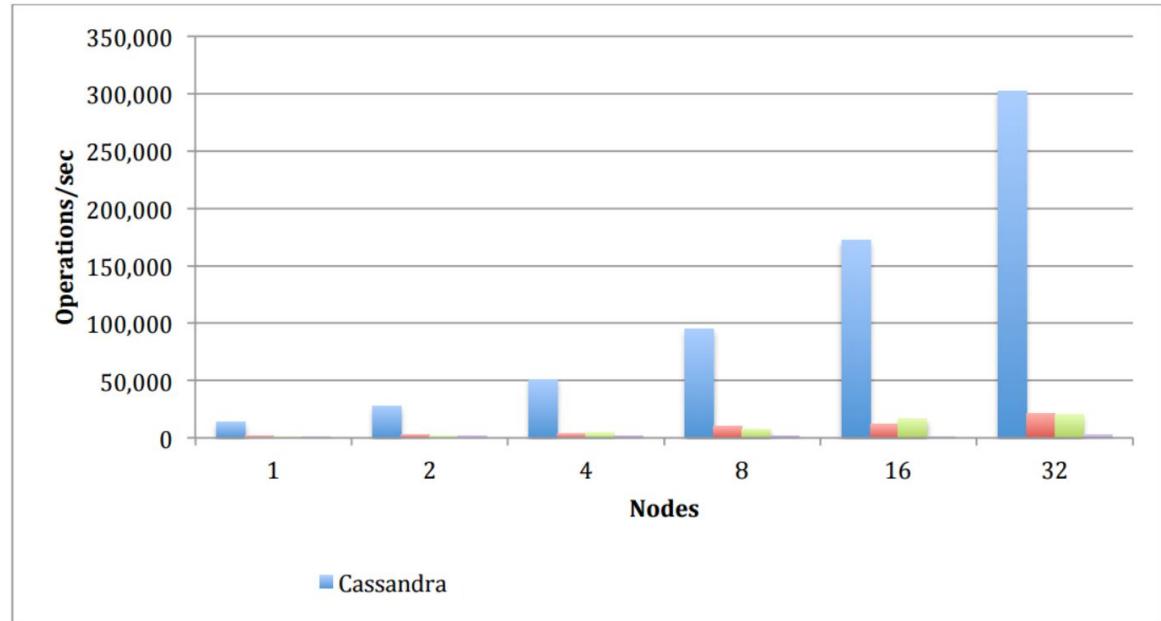
Apache Cassandra™ = NoSQL Distributed Database



Scales Linearly

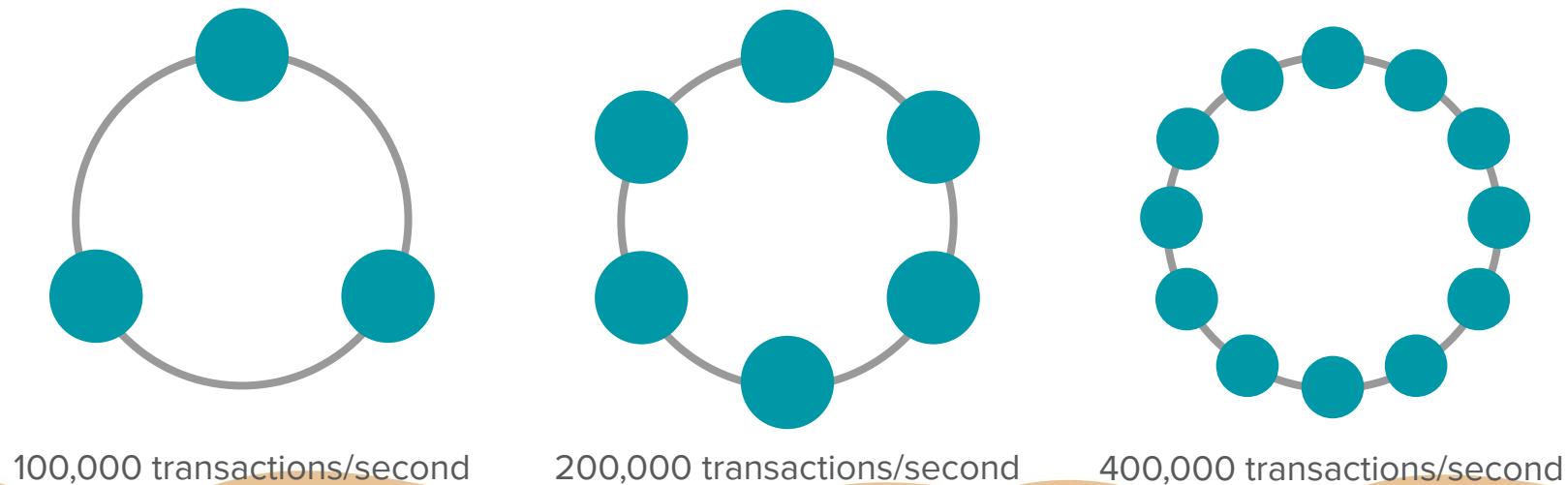
- Need more capacity?
- Need more throughput?
- Add nodes!

Balanced Read/Write Mix

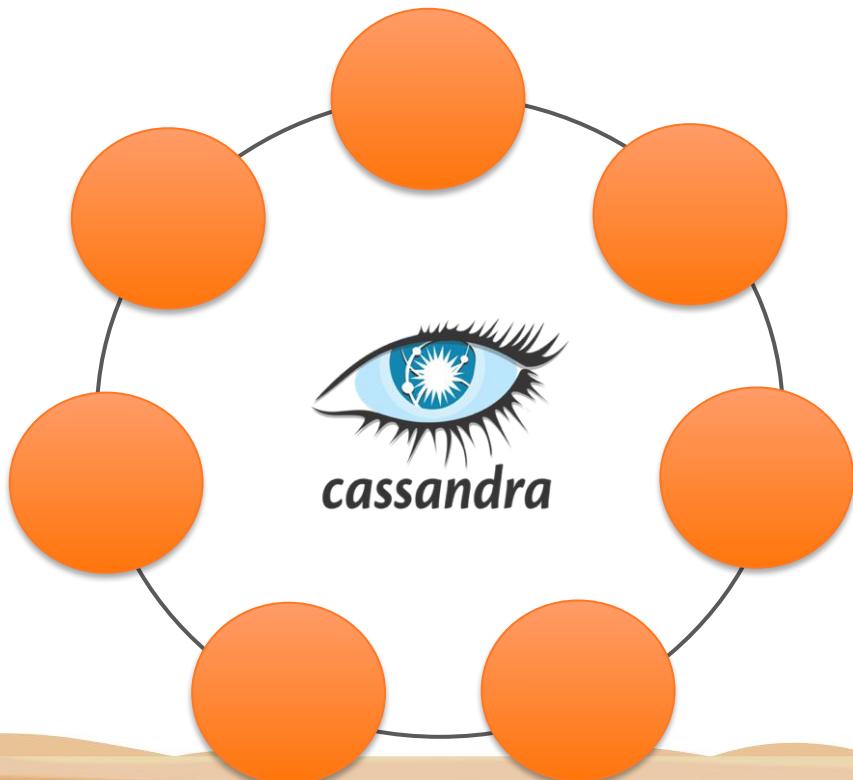


Horizontal vs. Vertical Scaling

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



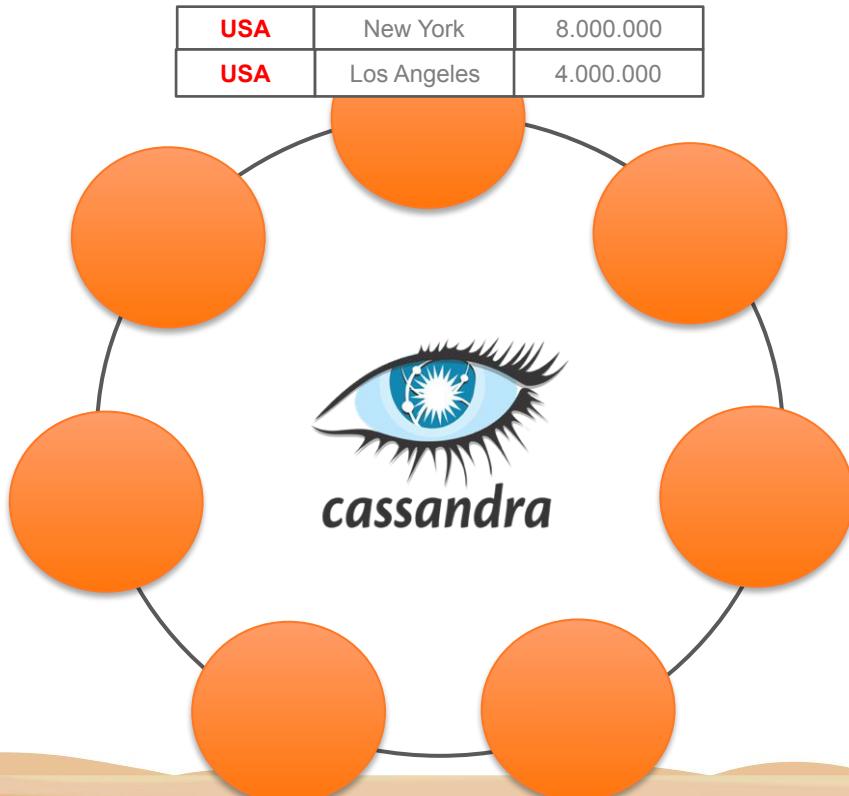
Data is Distributed



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

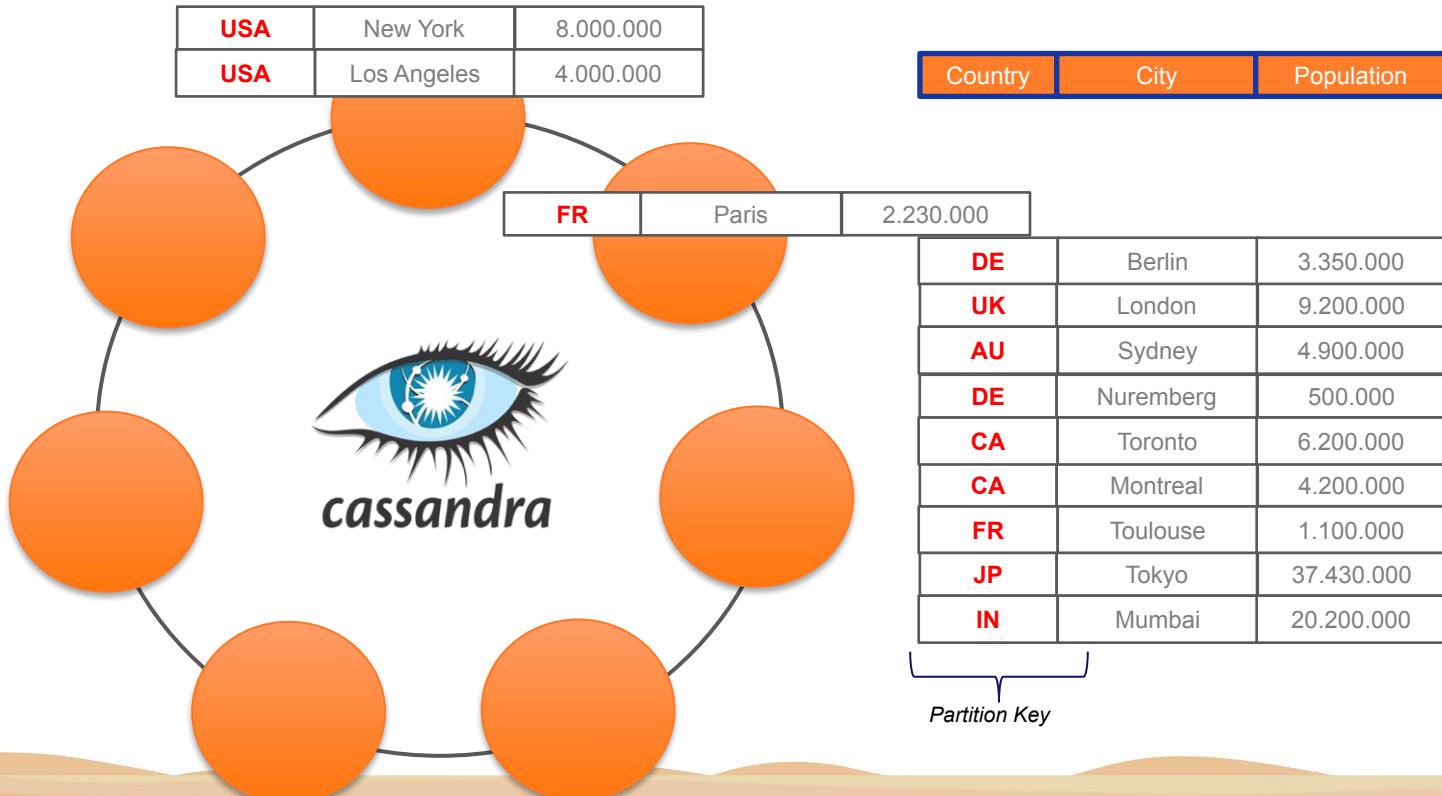
Data is Distributed



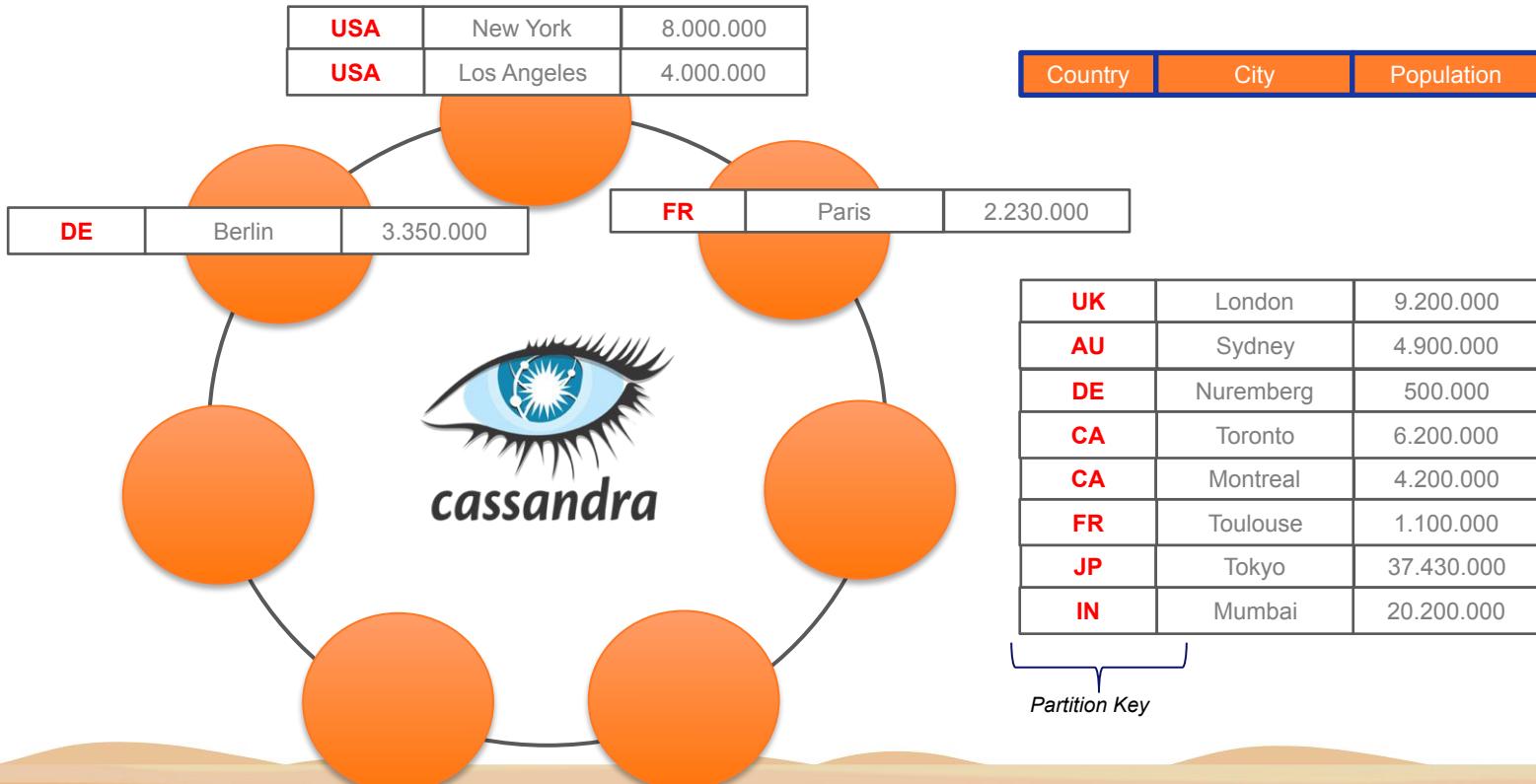
FR	Paris	2.230.000
DE	Berlin	3.350.000
UK	London	9.200.000
AU	Sydney	4.900.000
DE	Nuremberg	500.000
CA	Toronto	6.200.000
CA	Montreal	4.200.000
FR	Toulouse	1.100.000
JP	Tokyo	37.430.000
IN	Mumbai	20.200.000

Partition Key

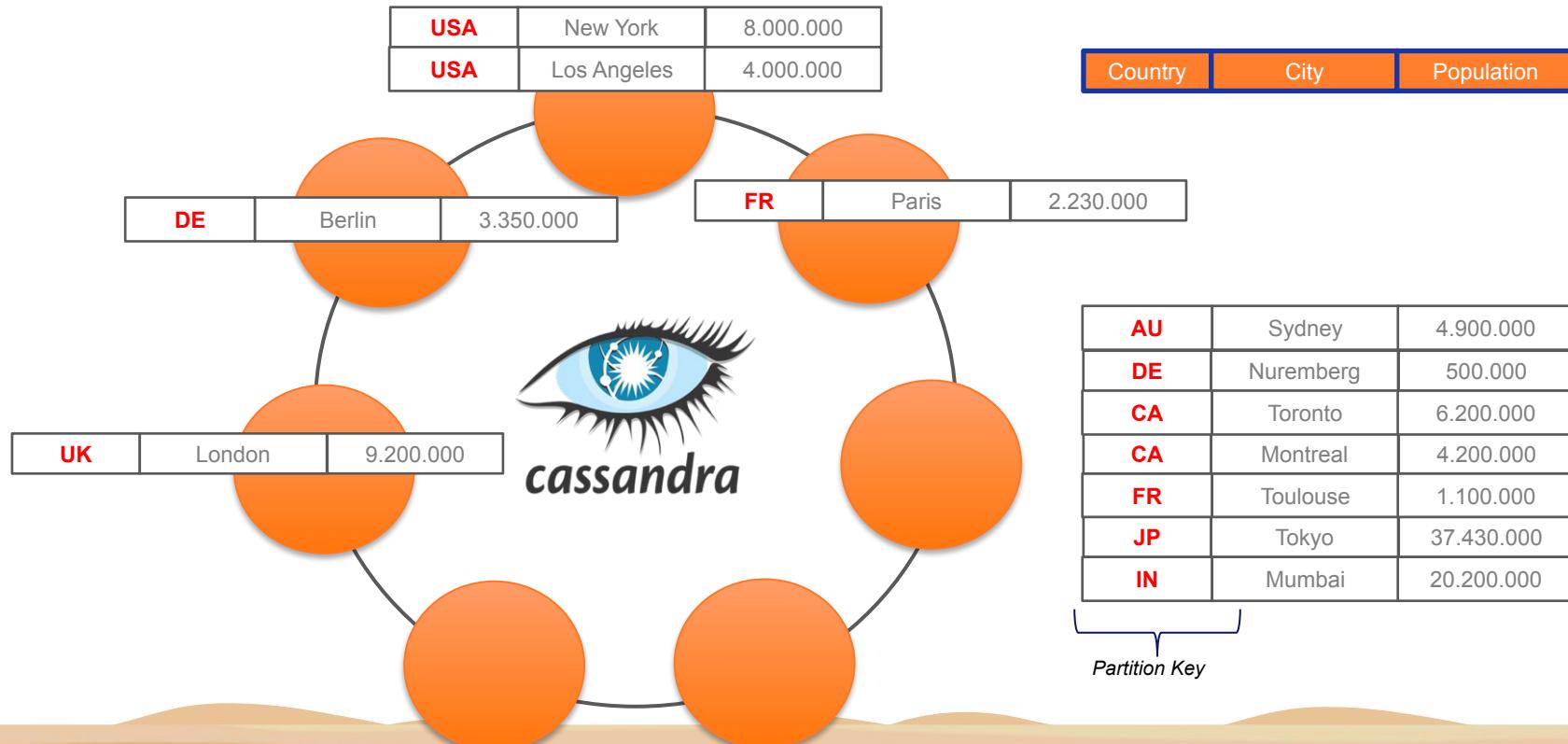
Data is Distributed



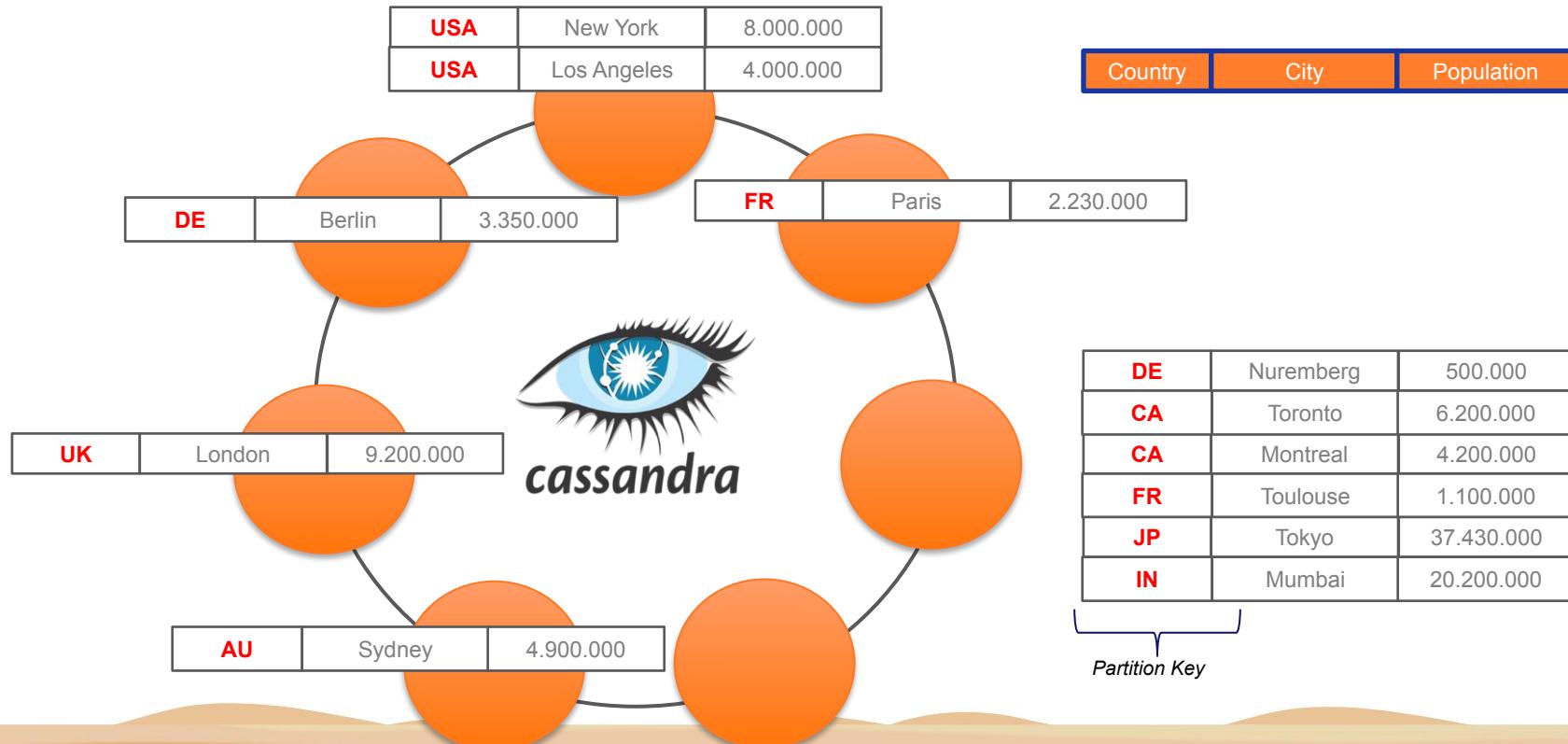
Data is Distributed



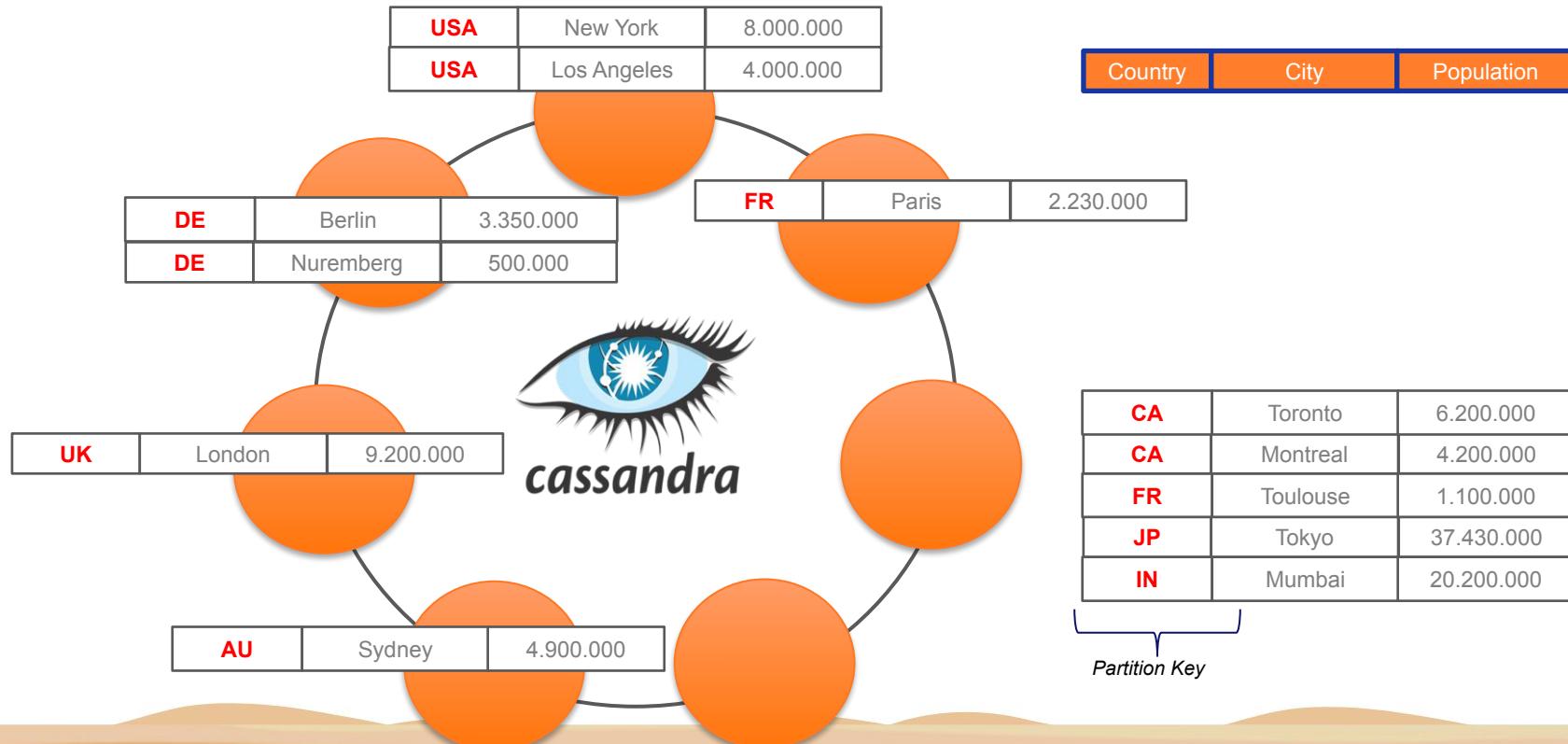
Data is Distributed



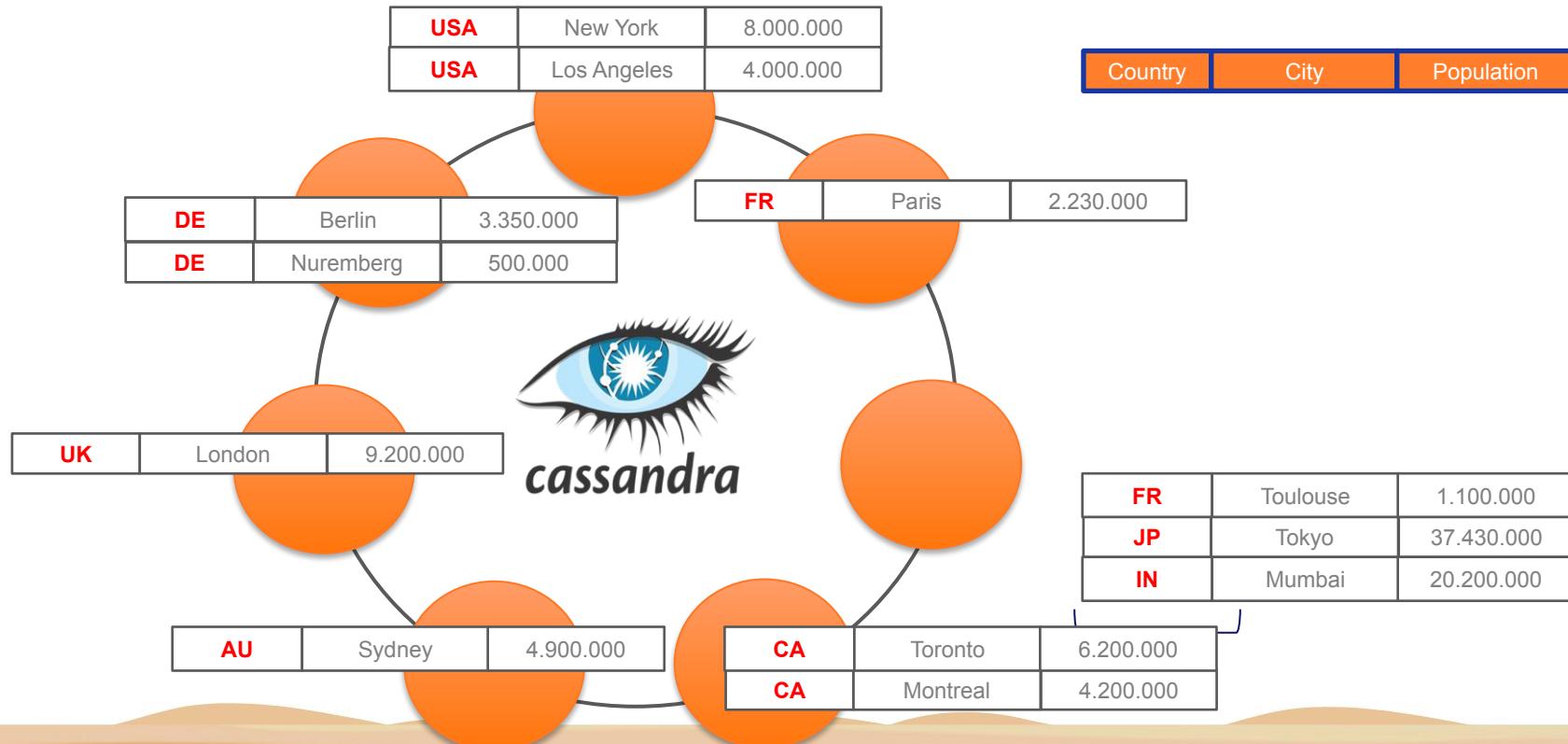
Data is Distributed



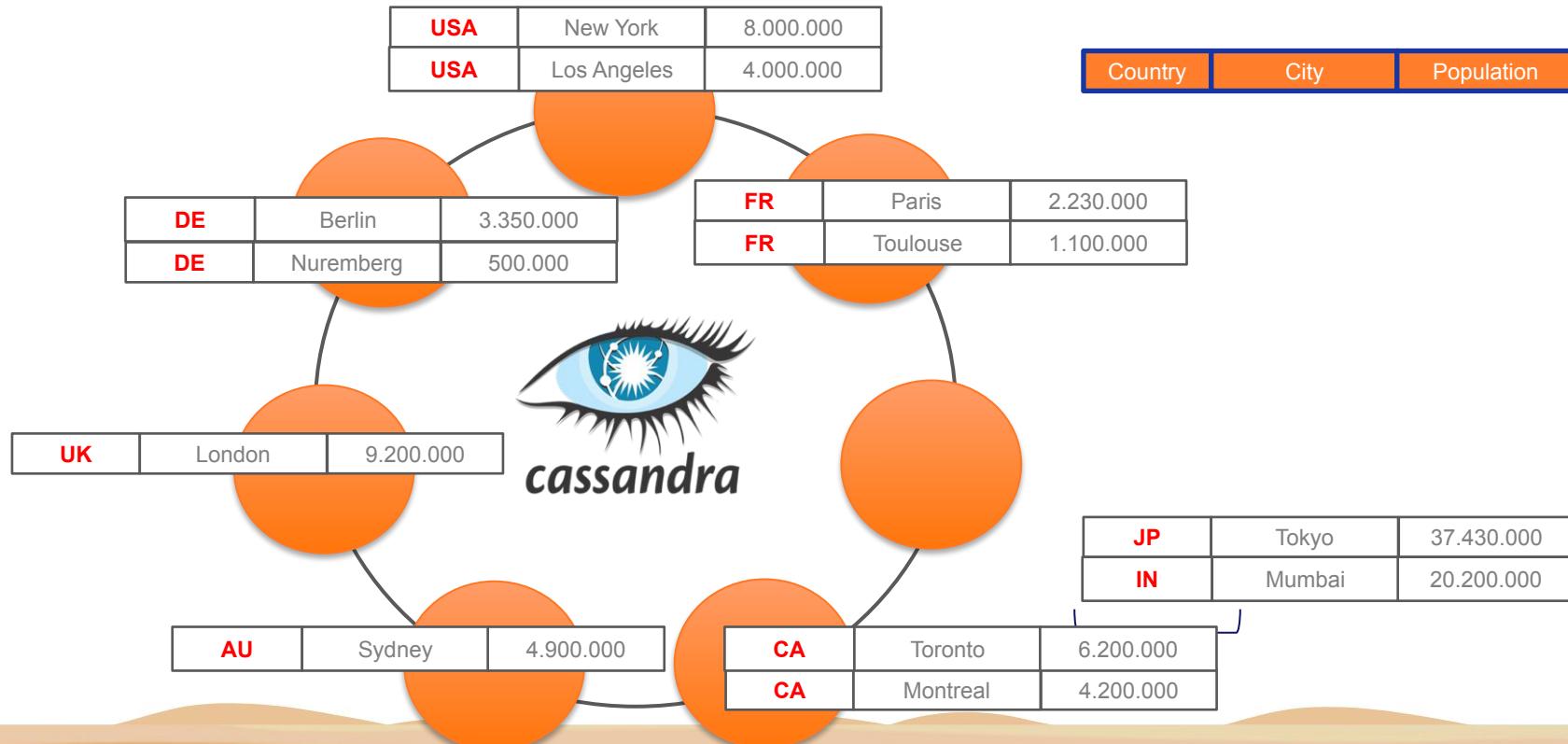
Data is Distributed



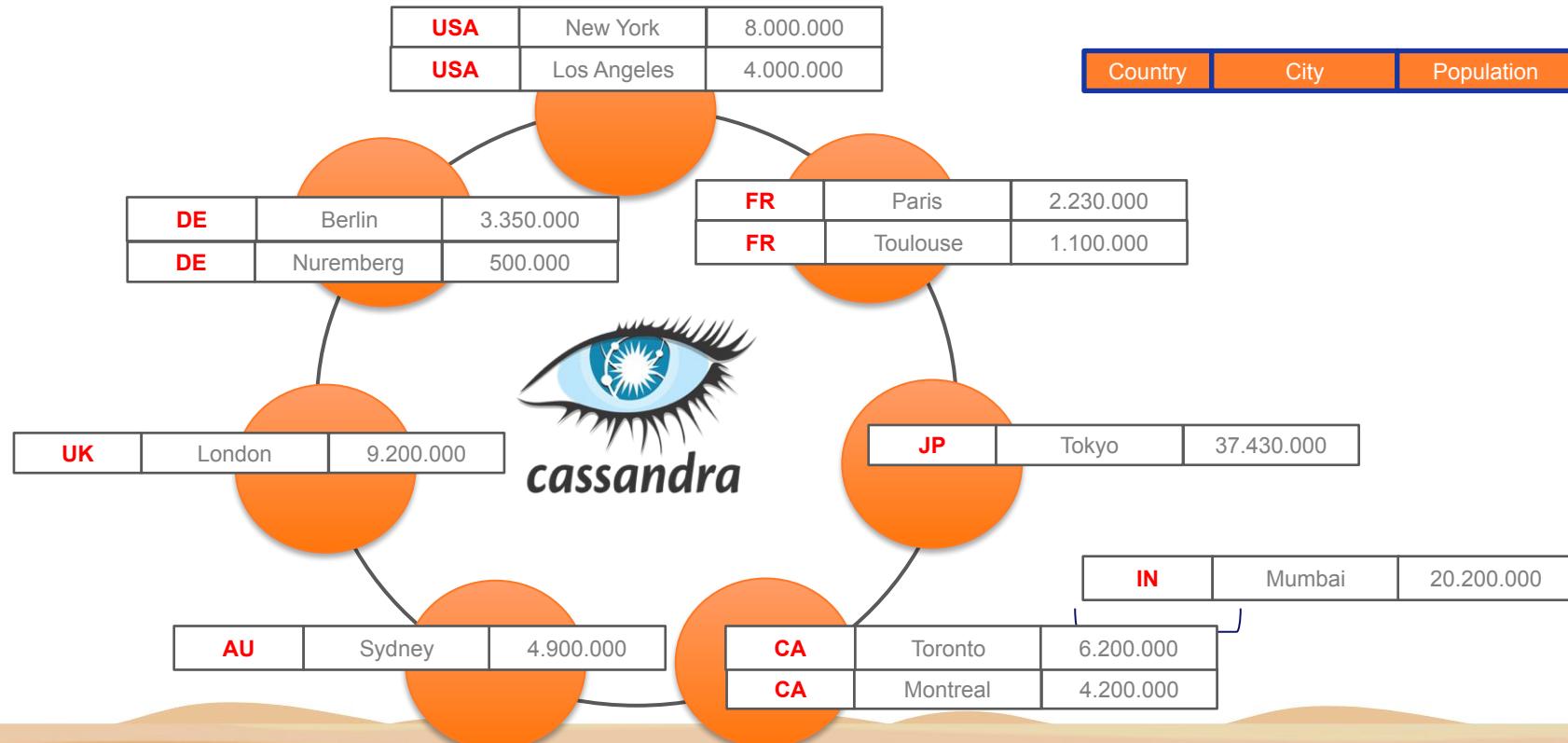
Data is Distributed



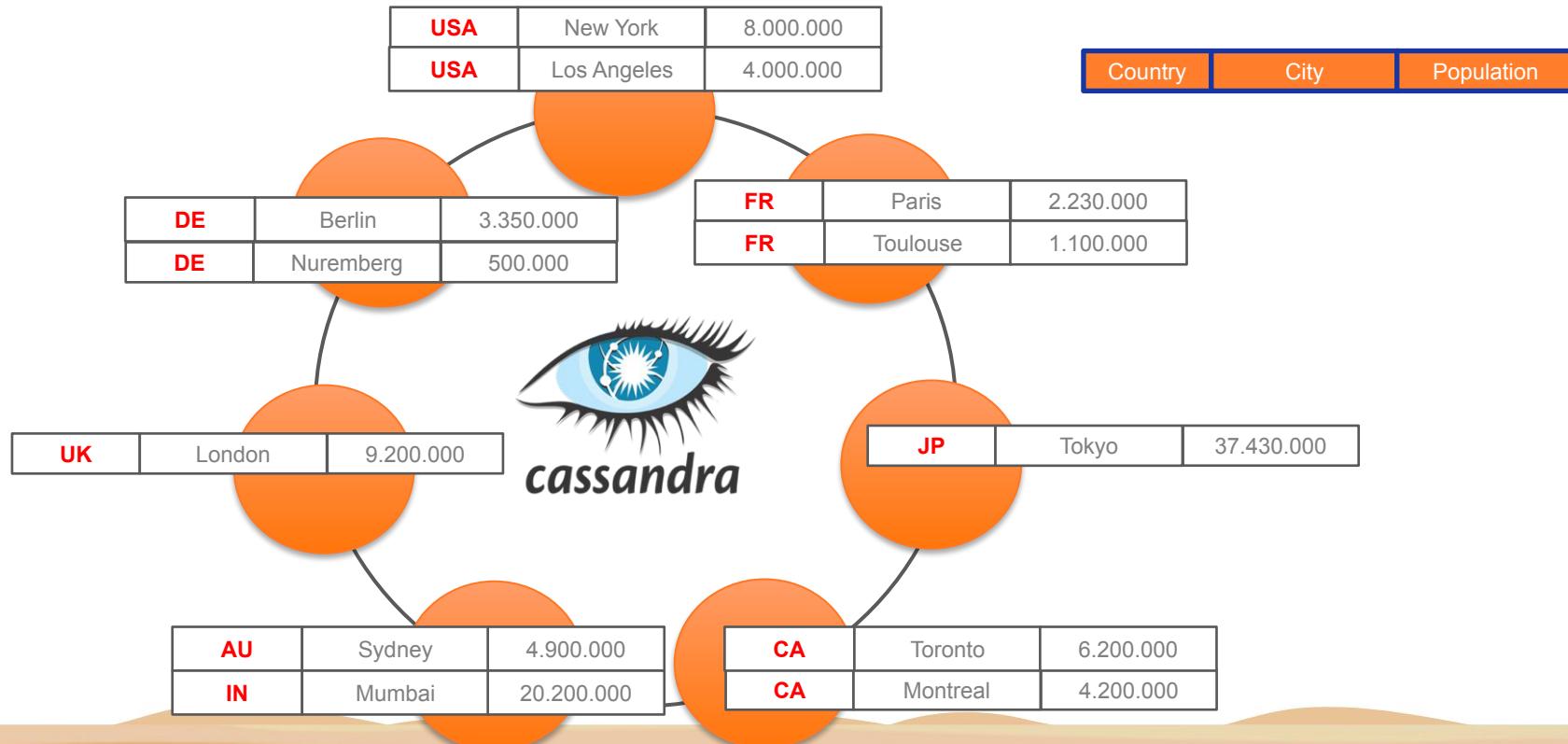
Data is Distributed



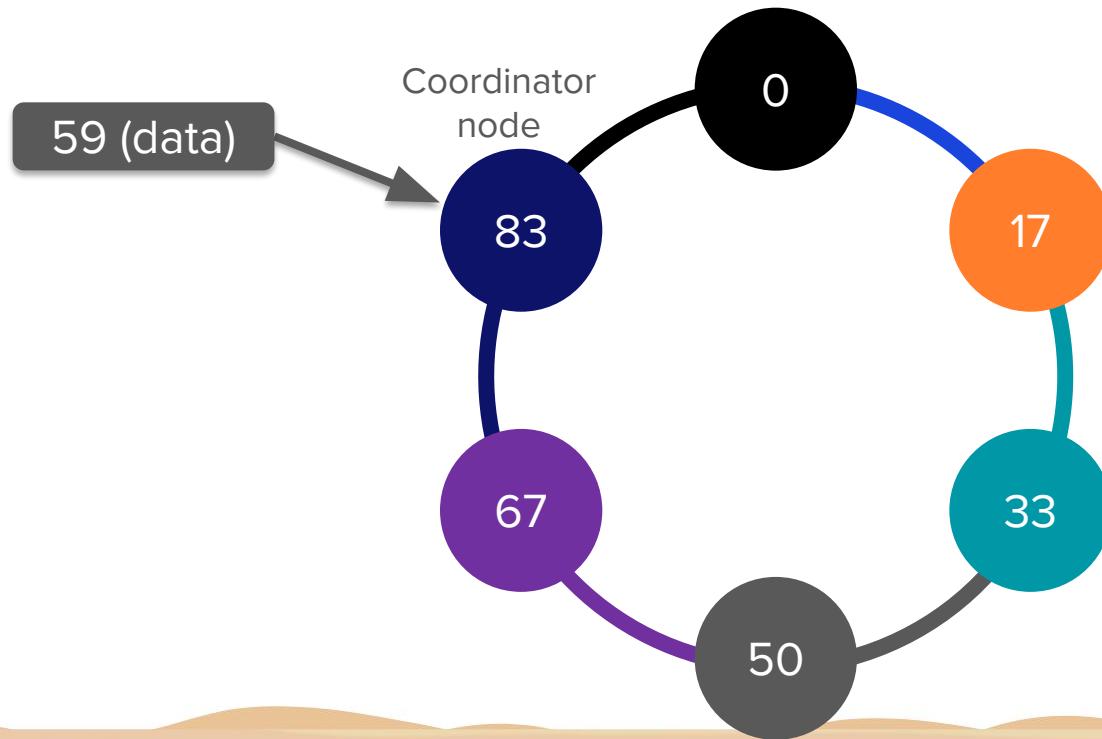
Data is Distributed



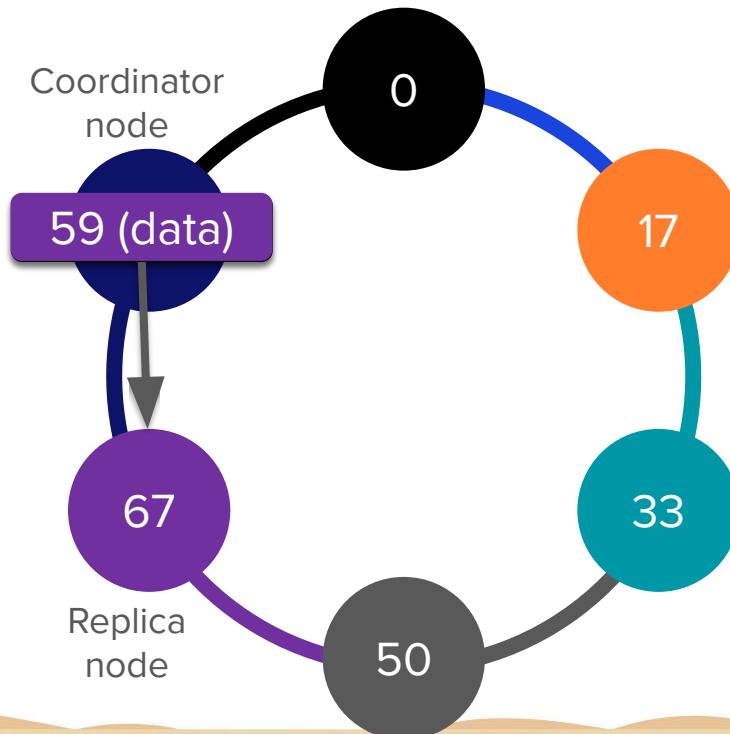
Data is Distributed



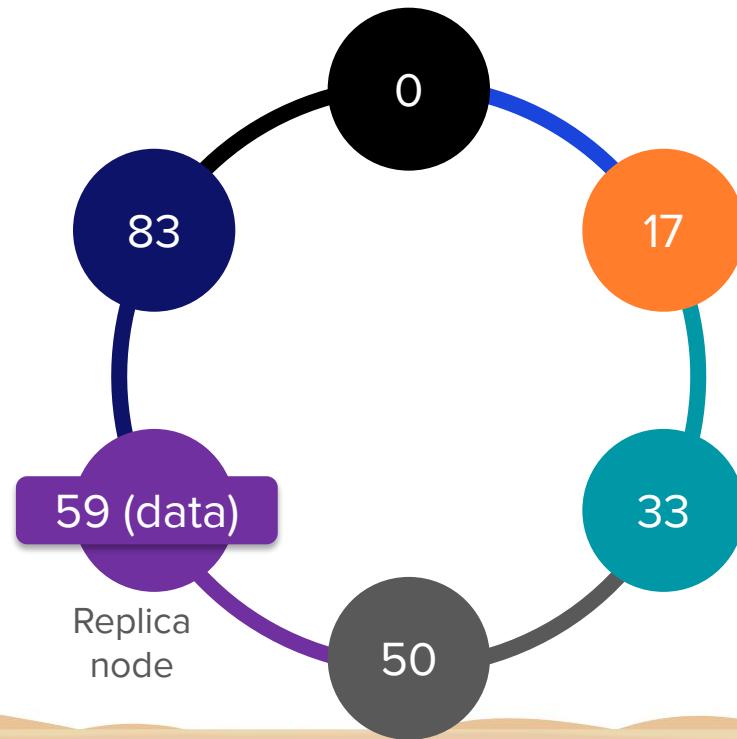
How the Ring Works



How the Ring Works

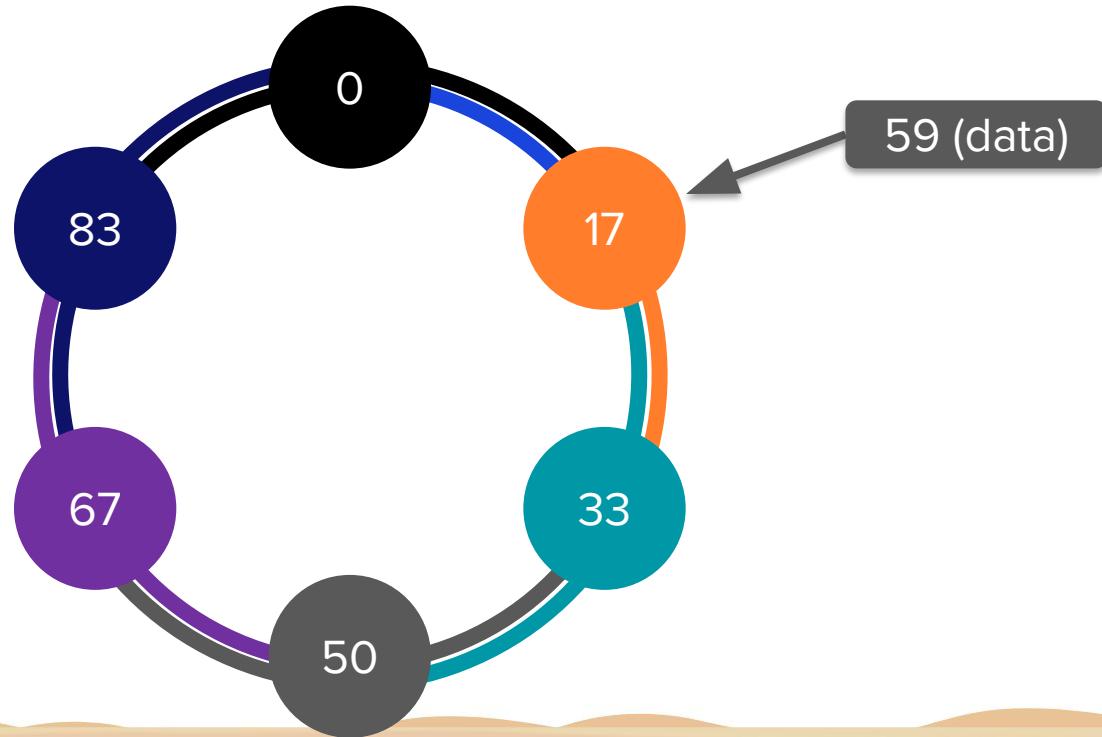


How the Ring Works



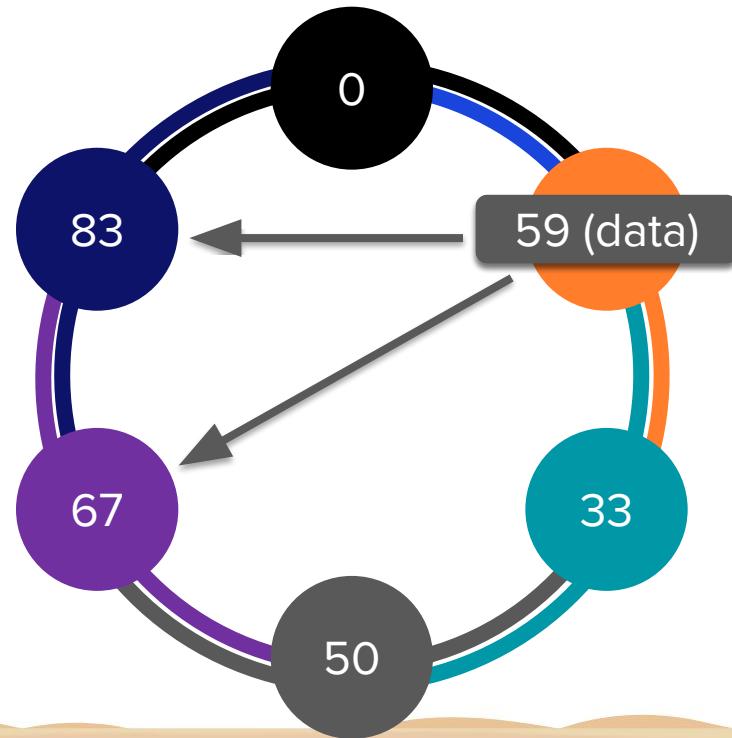
Replication within the Ring

RF = 2



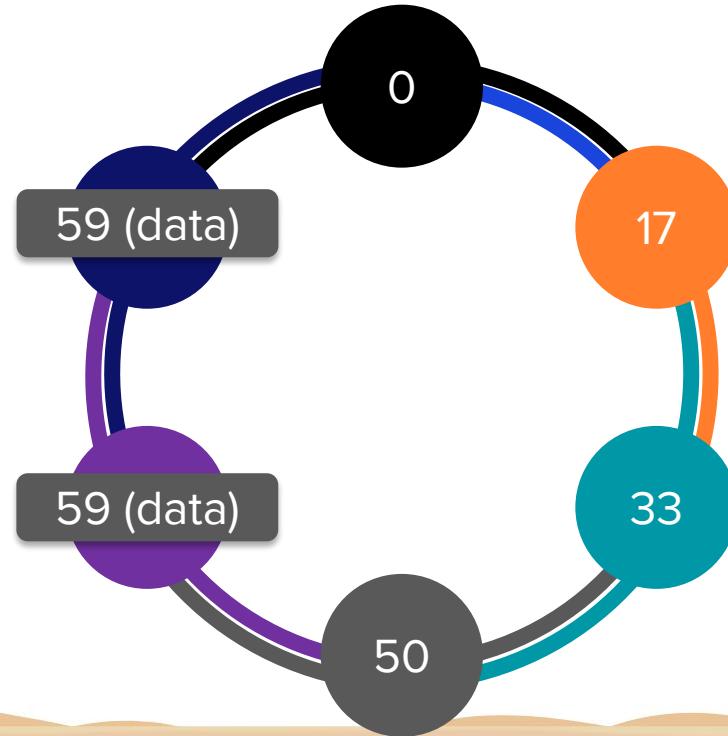
Replication within the Ring

RF = 2



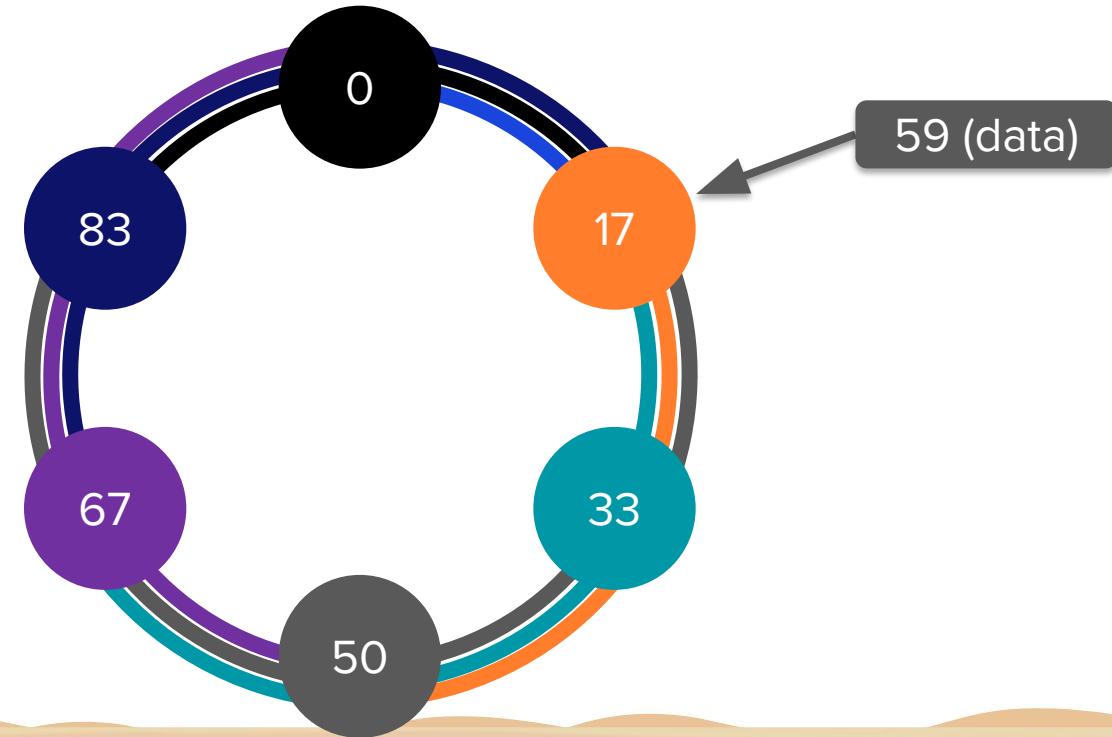
Replication within the Ring

RF = 2



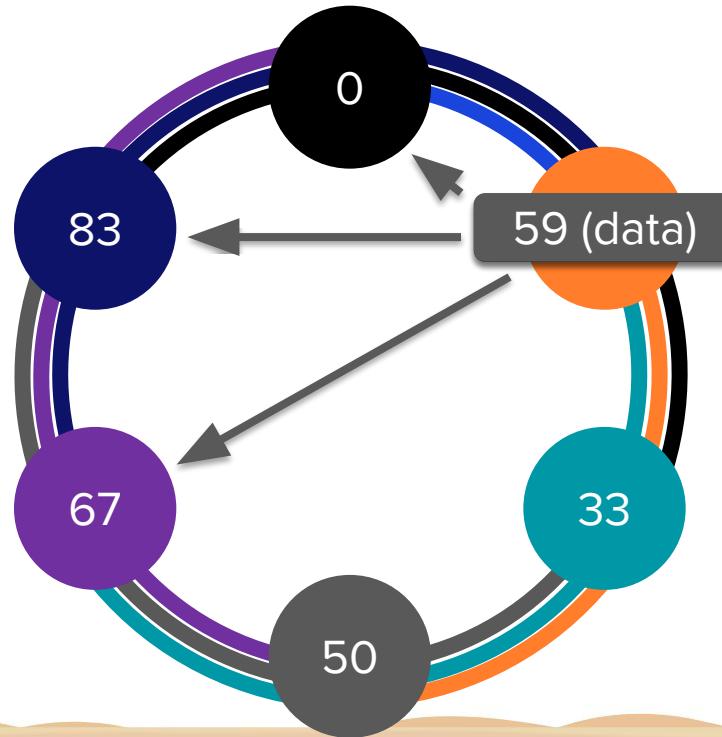
Replication within the Ring

RF = 3



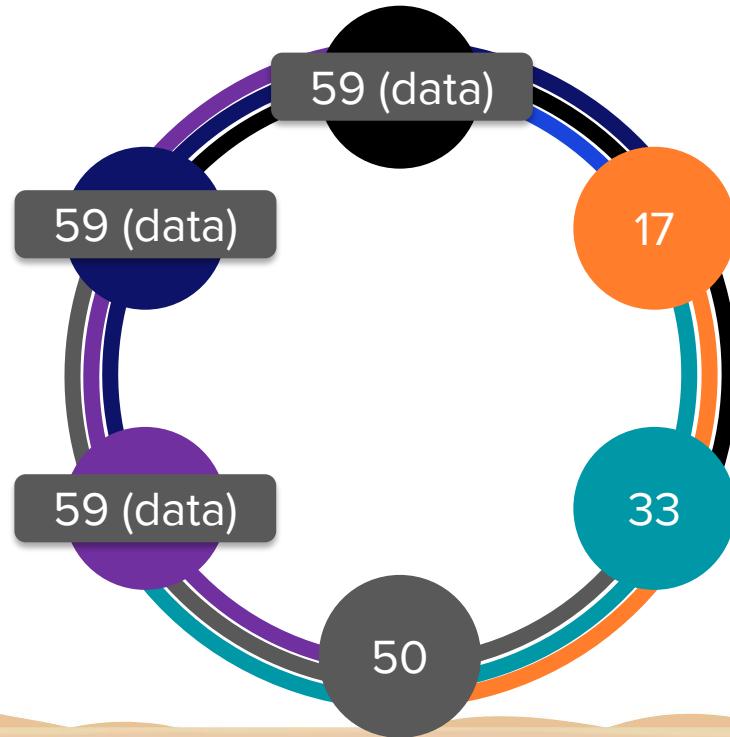
Replication within the Ring

RF = 3



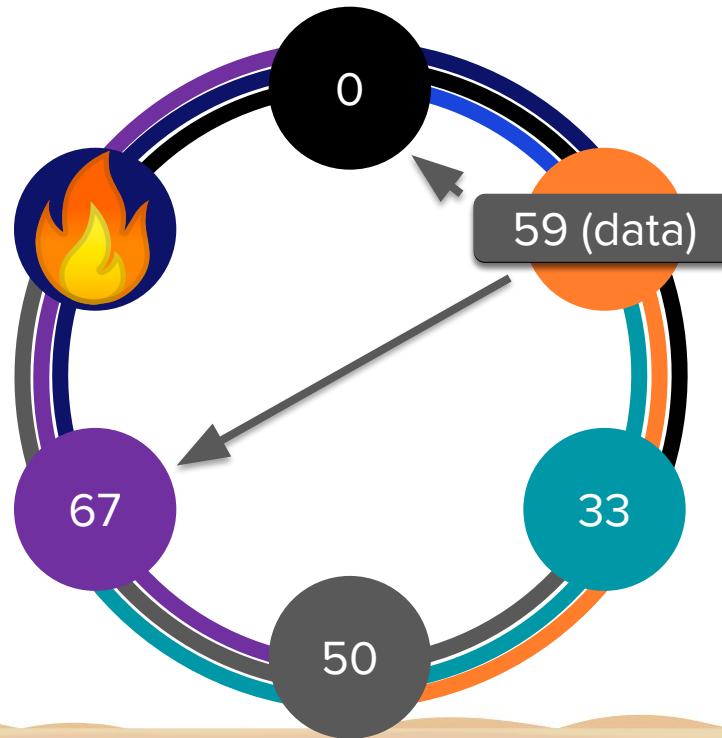
Replication within the Ring

RF = 3



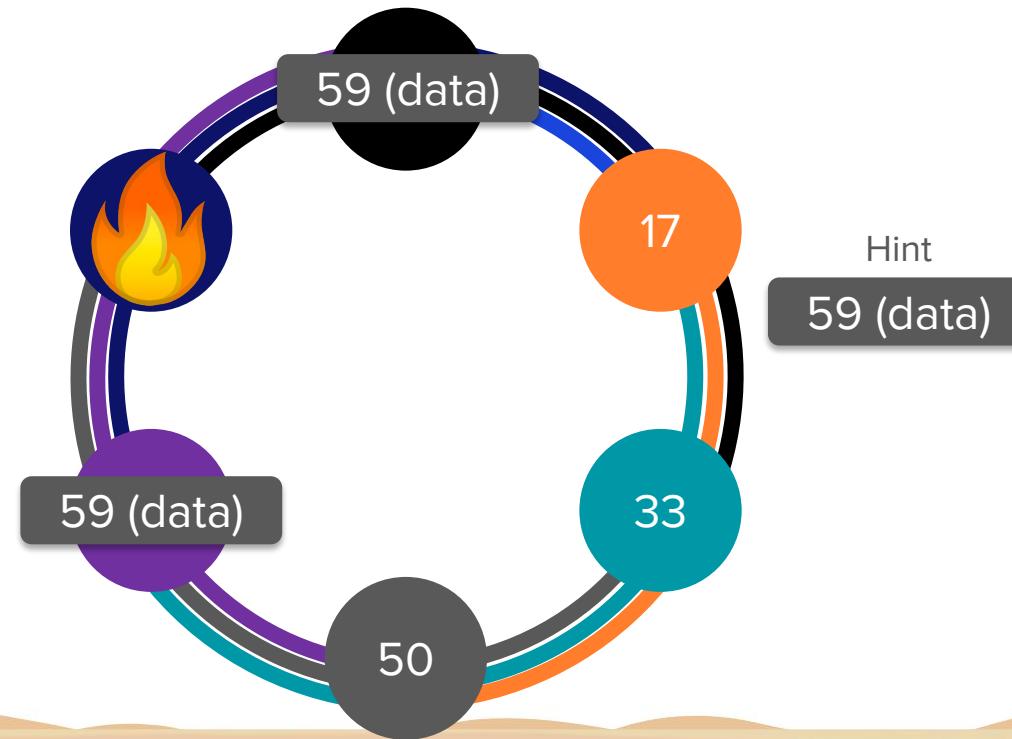
Node Failure

RF = 3



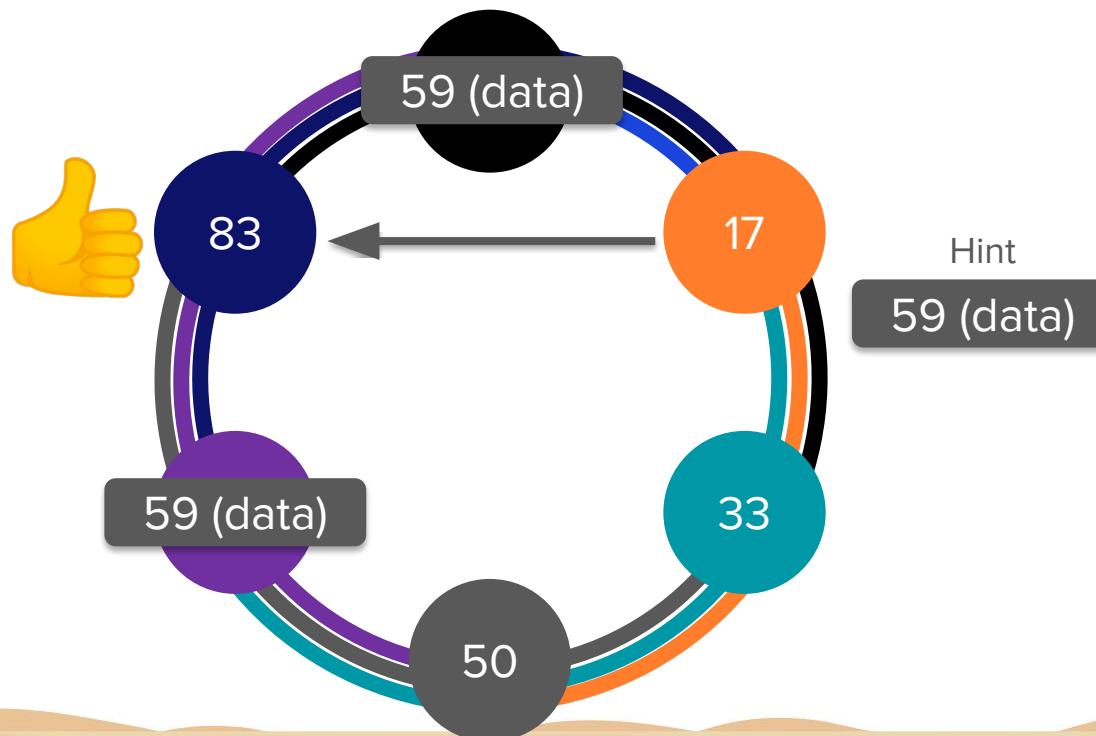
Node Failure

RF = 3



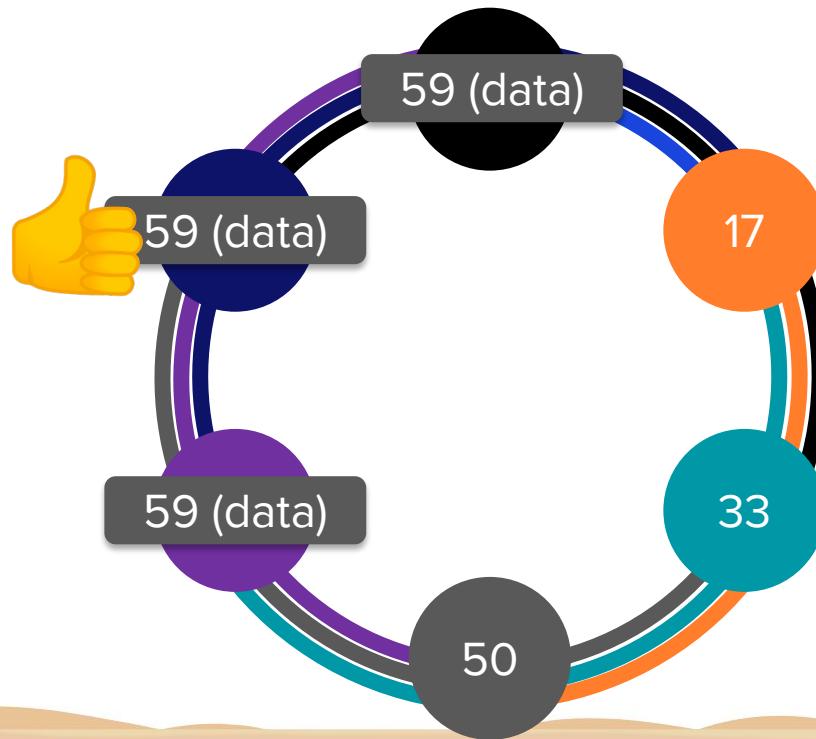
Node Failure

RF = 3



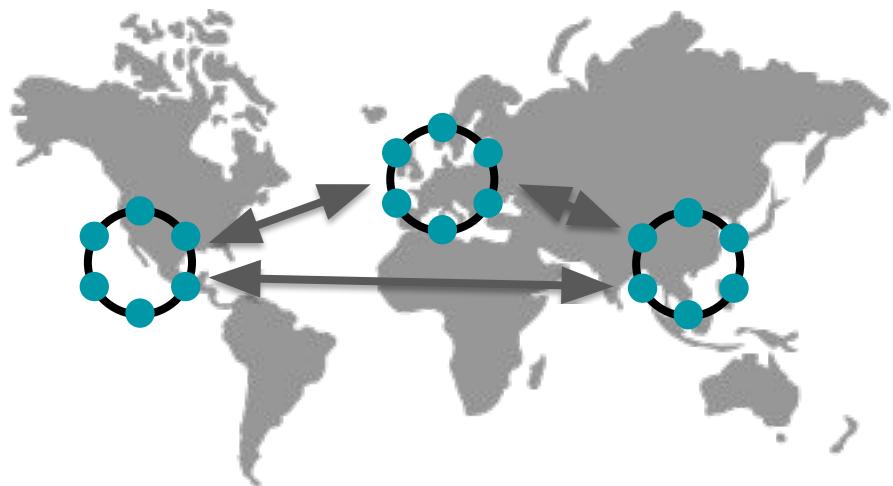
Node Failure – Recovered!

RF = 3

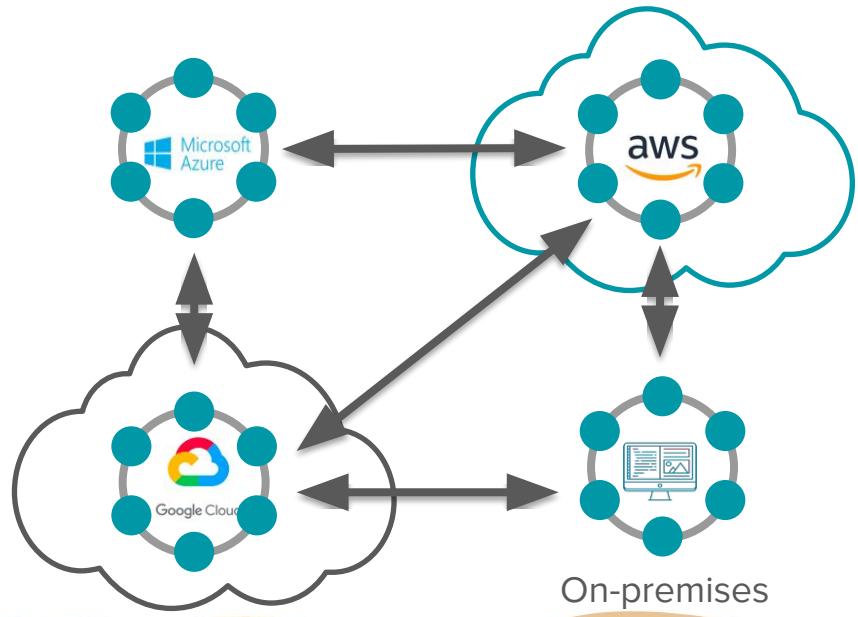


Data Distributed Everywhere

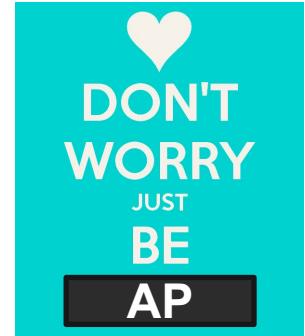
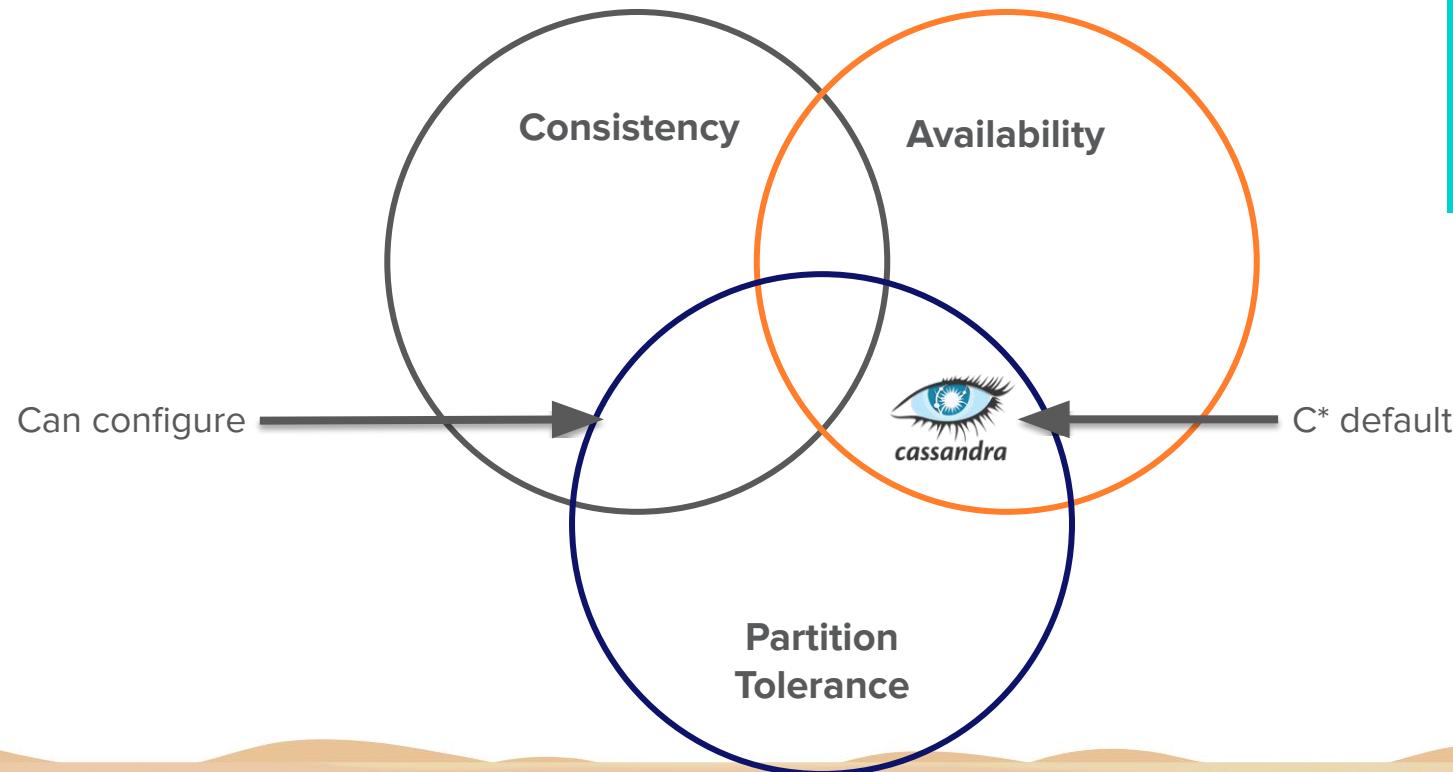
- Geographic Distribution



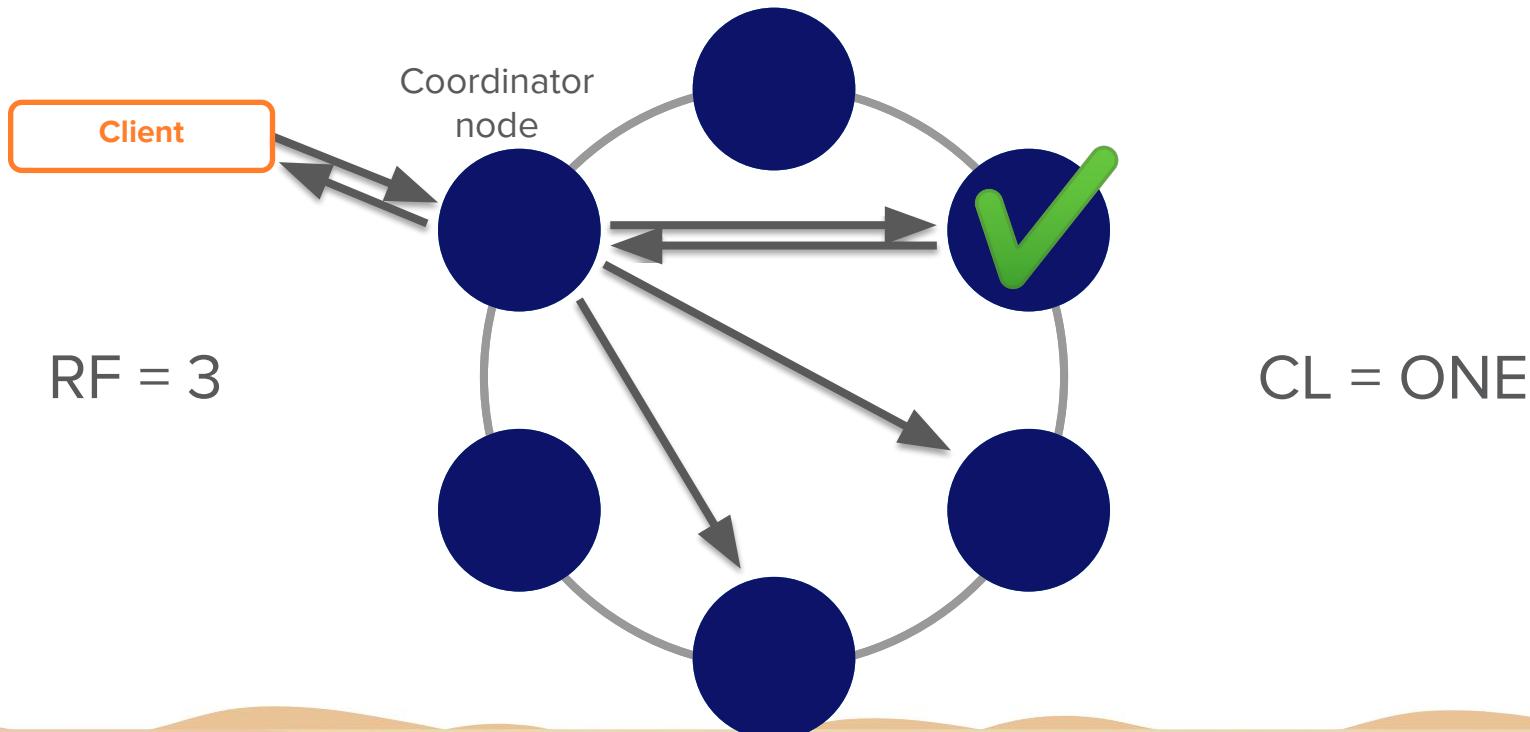
- Hybrid-Cloud and Multi-Cloud



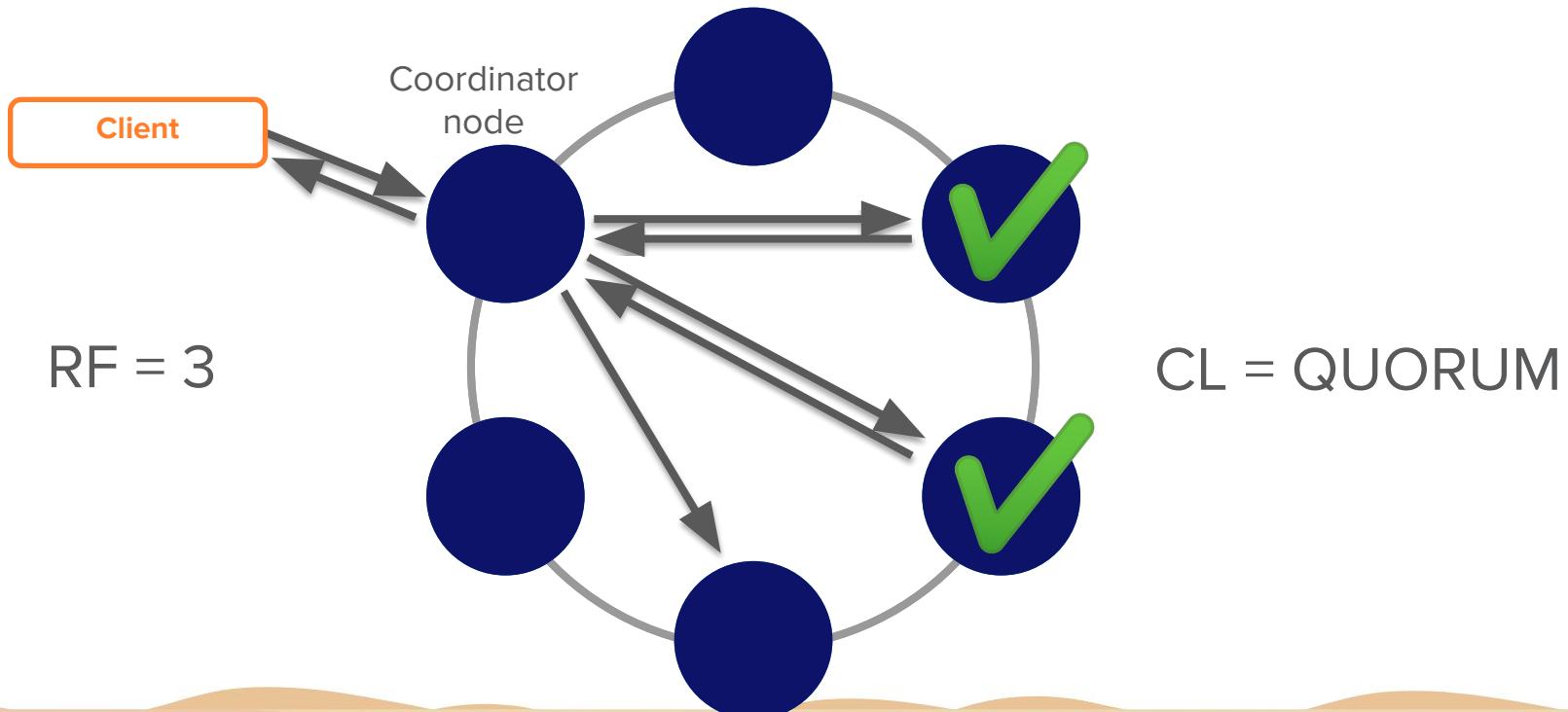
Cap Theorem



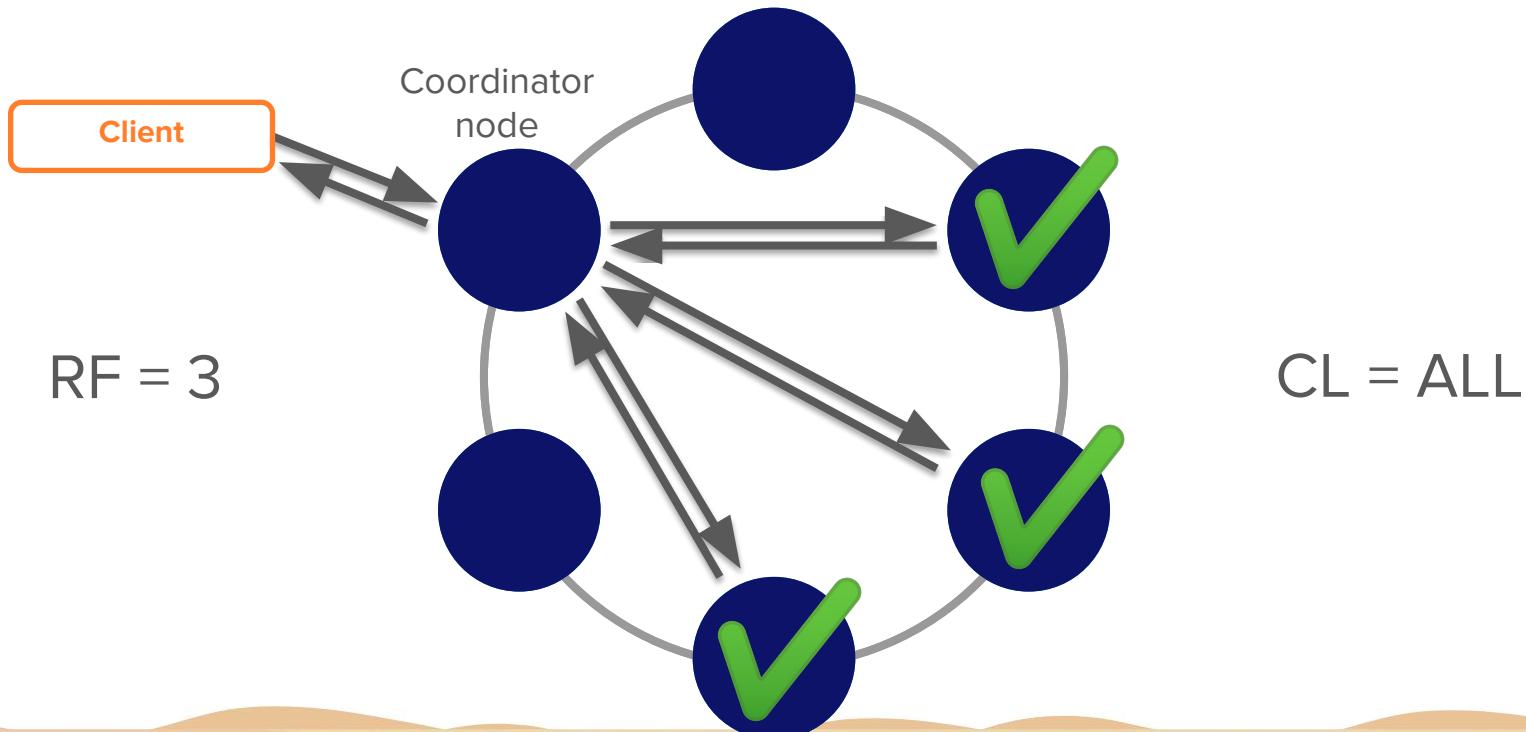
Consistency Levels



Consistency Levels



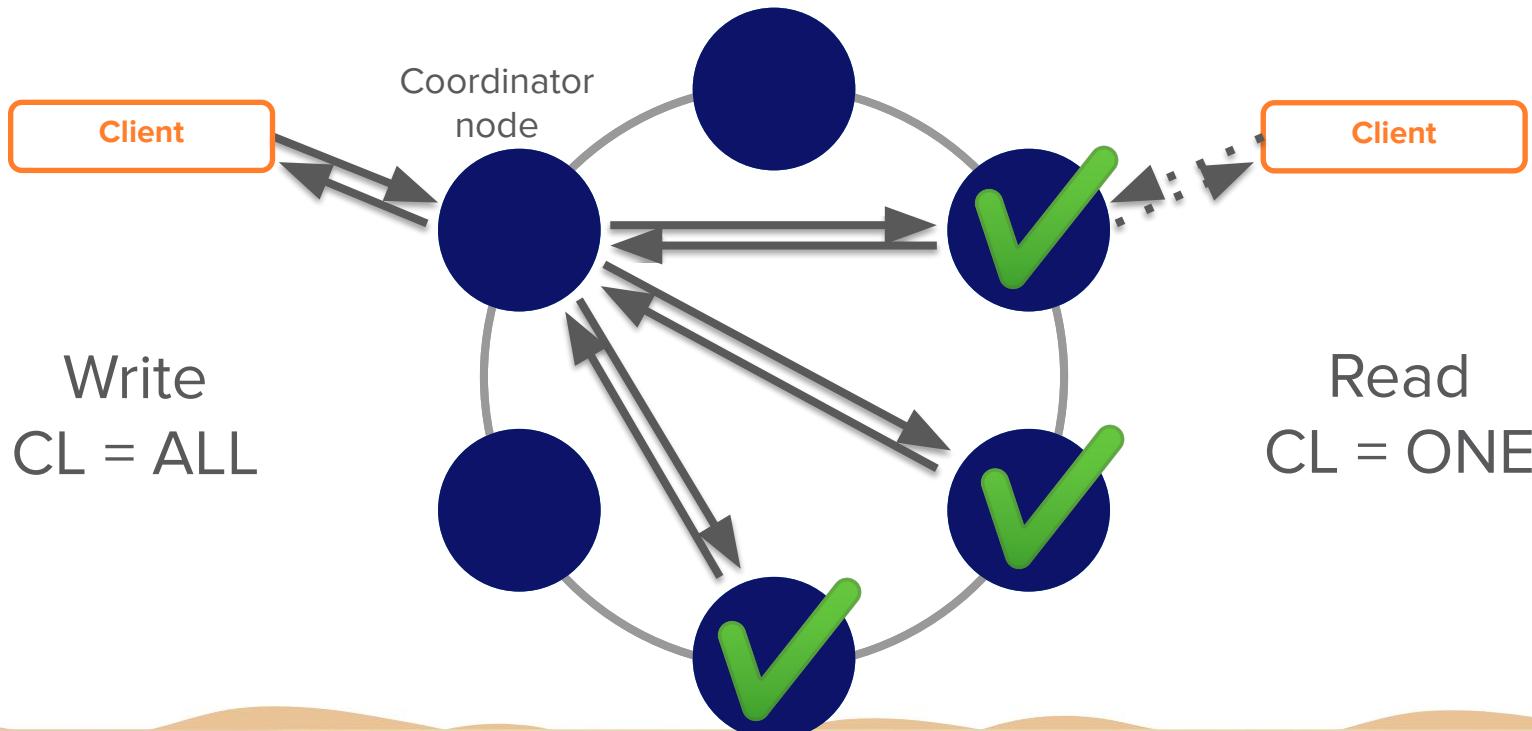
Consistency Levels



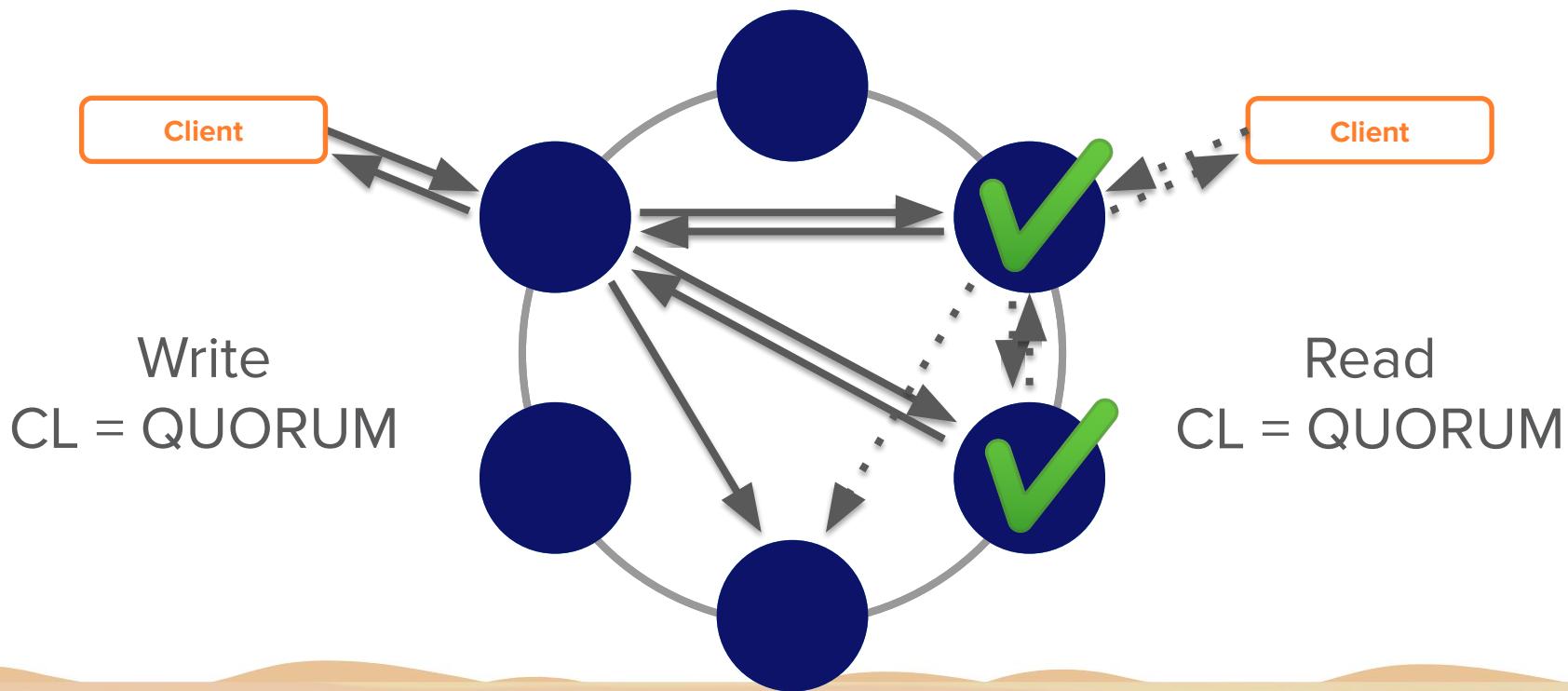
Immediate Consistency


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

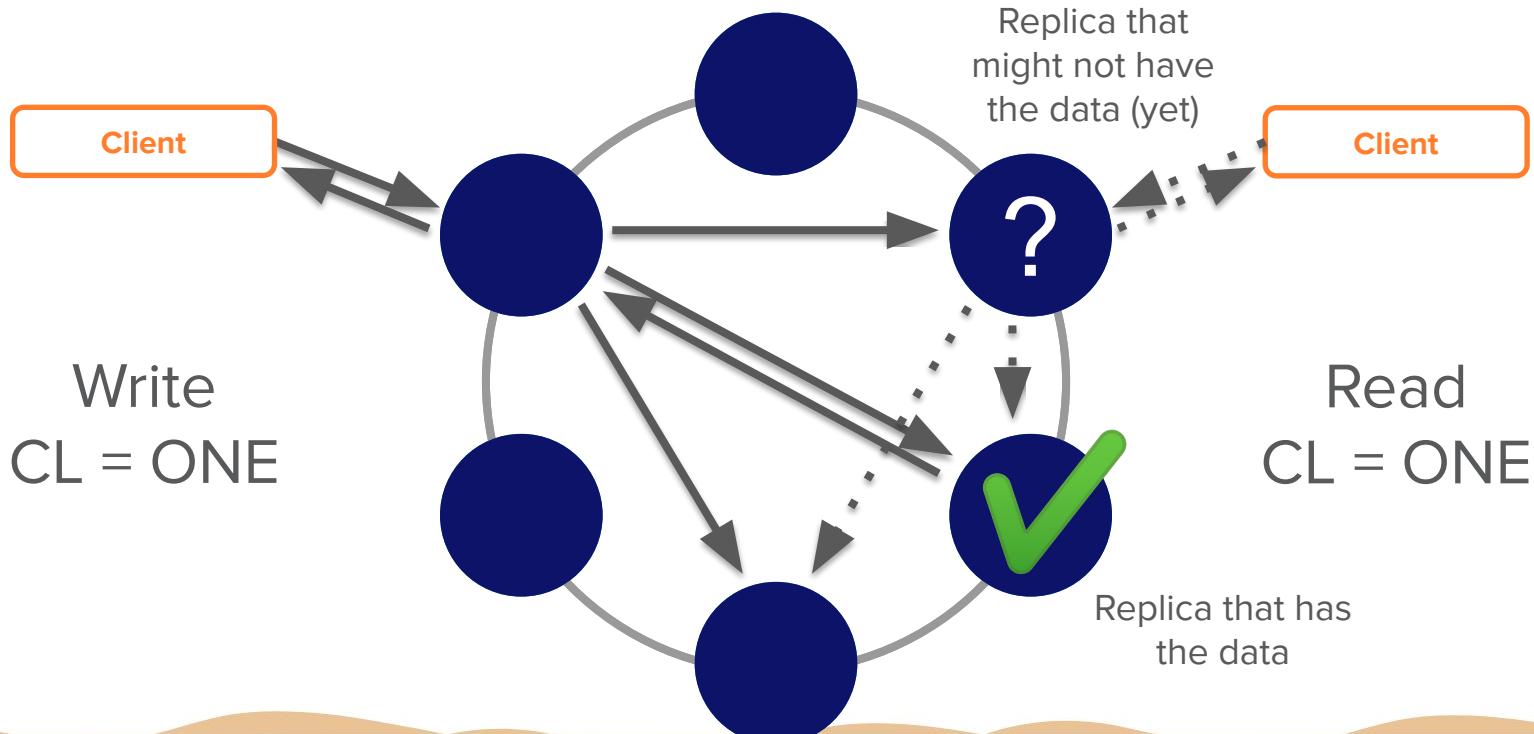
Immediate Consistency – One Way



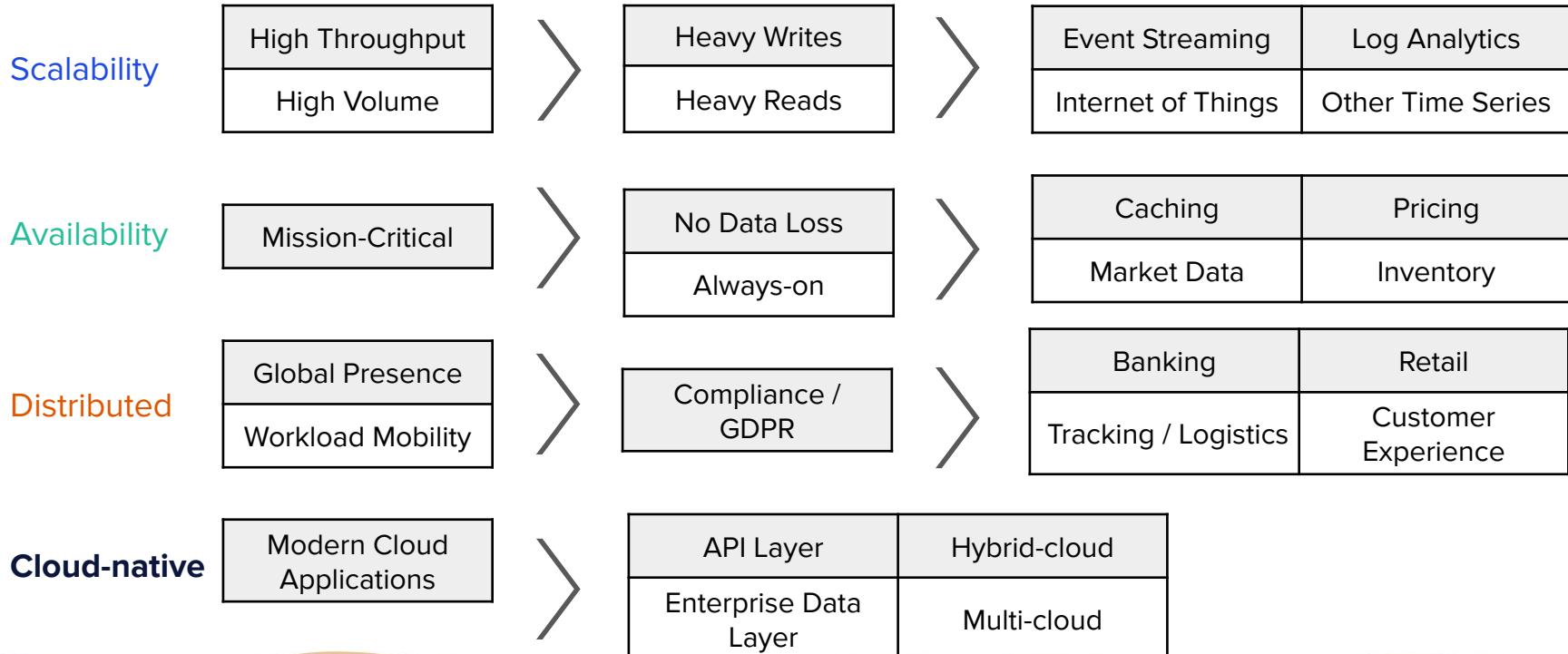
Immediate Consistency – A Better Way



Weak(er) Consistency



Understanding Use Cases



Exercise

DataStax Studio



Cassandra Developer Workshop #2 - Datastax Studio

Use the Markdown Editor

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.

Language: **Markdown**

Click here to see the code editor

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.

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

RAM

HDD

Write Path

RAM

1 Dev Awesome

TX

Houston

HDD



Write Path

MemTable

RAM

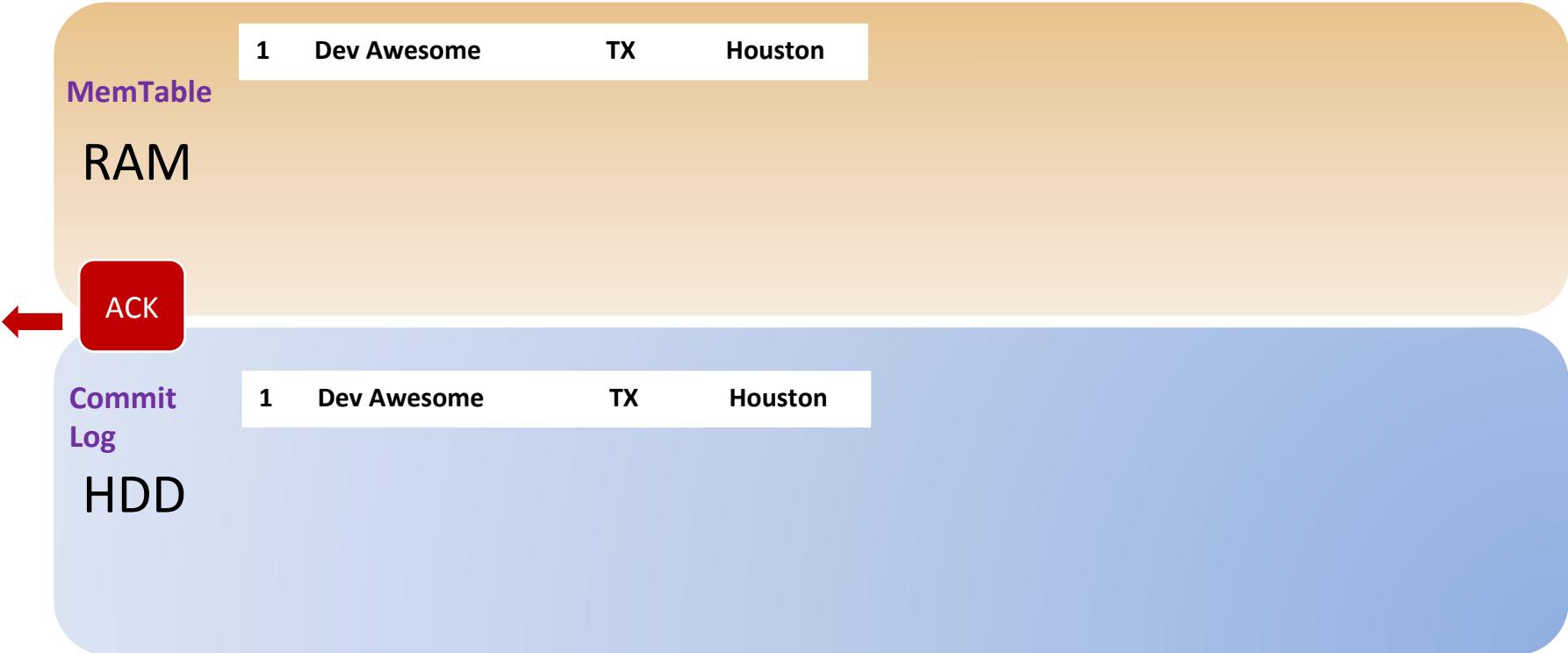
1	Dev Awesome	TX	Houston
---	-------------	----	---------

Commit Log

HDD

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

Write Path



Write Path

MemTable

RAM

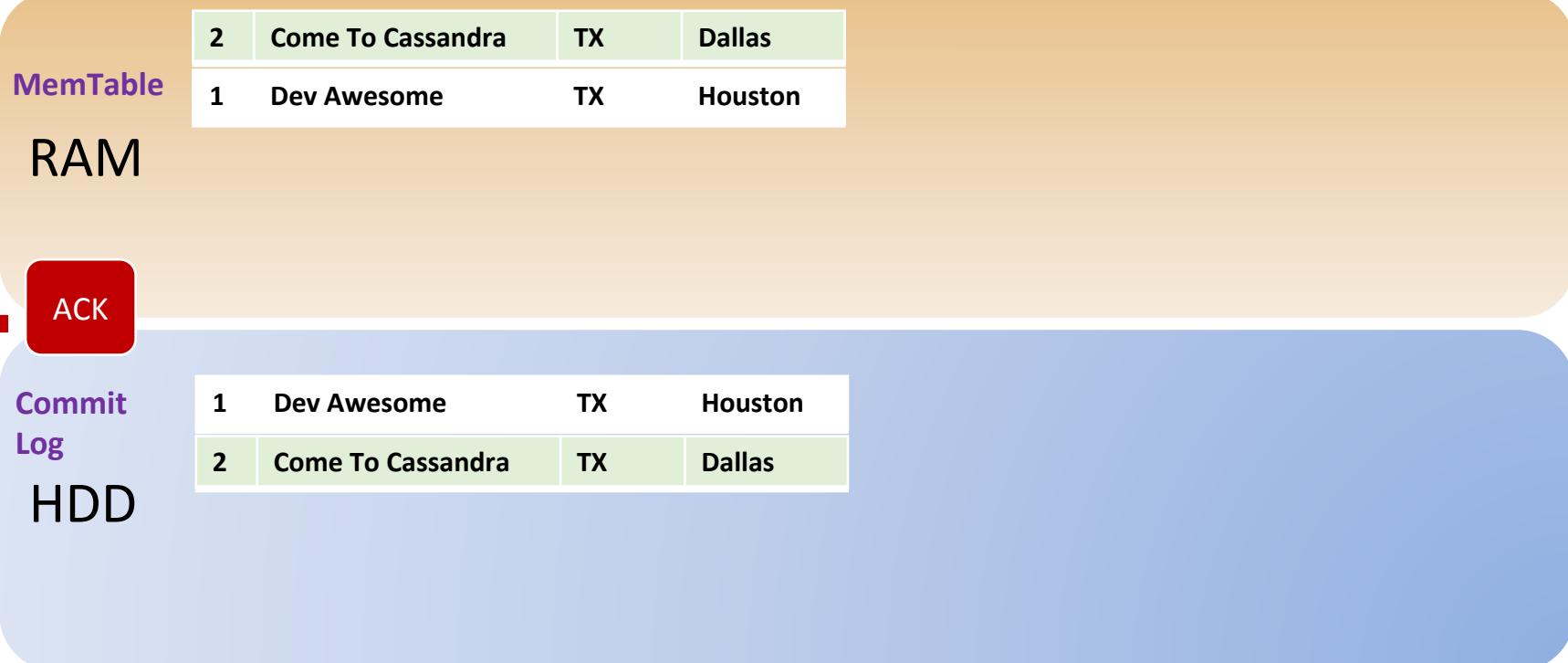
1 Dev Awesome TX Houston

2 Come To Cassandra TX Dallas

Commit Log 1 Dev Awesome TX Houston

HDD

Write Path



Write Path

MemTable

2	Come To Cassandra	TX	Dallas
1	Dev Awesome	TX	Houston

RAM

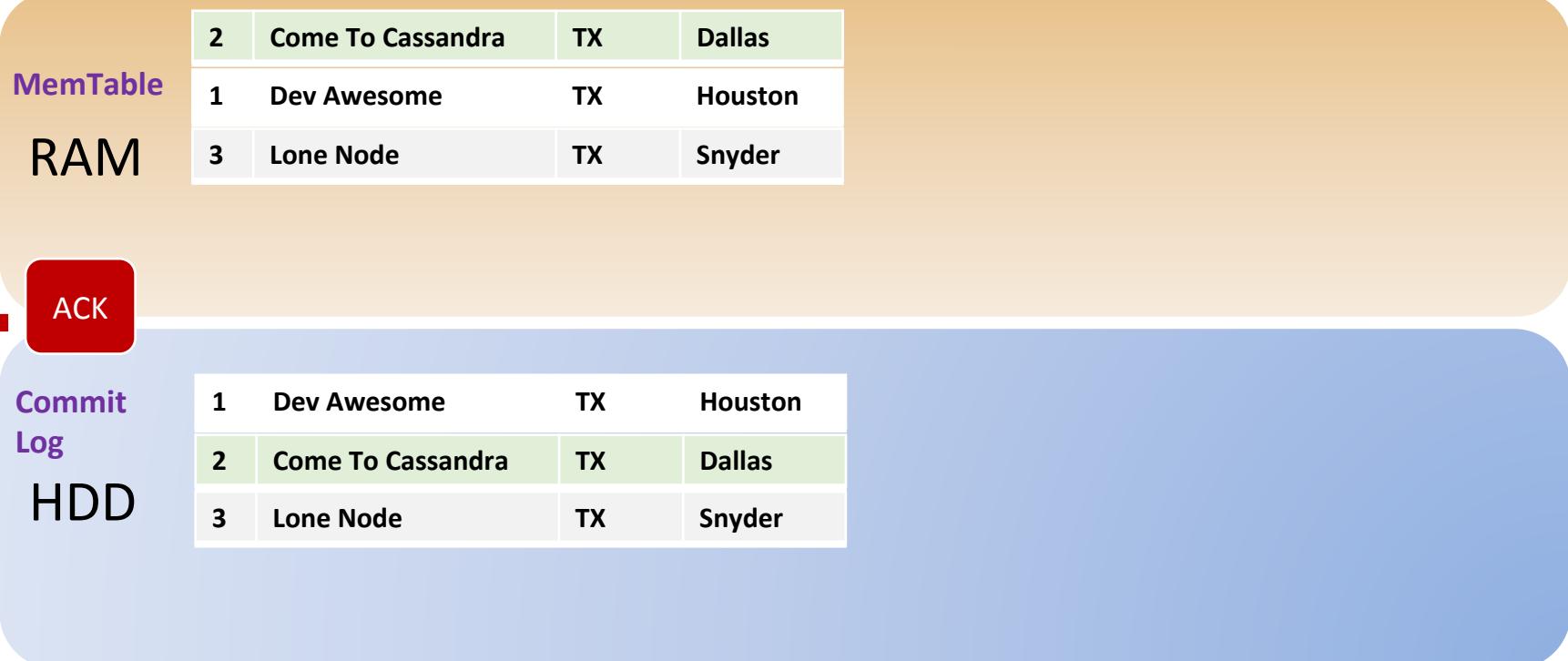


Commit
Log

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

HDD

Write Path



Write Path

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

→

Commit Log HDD	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
HDD

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



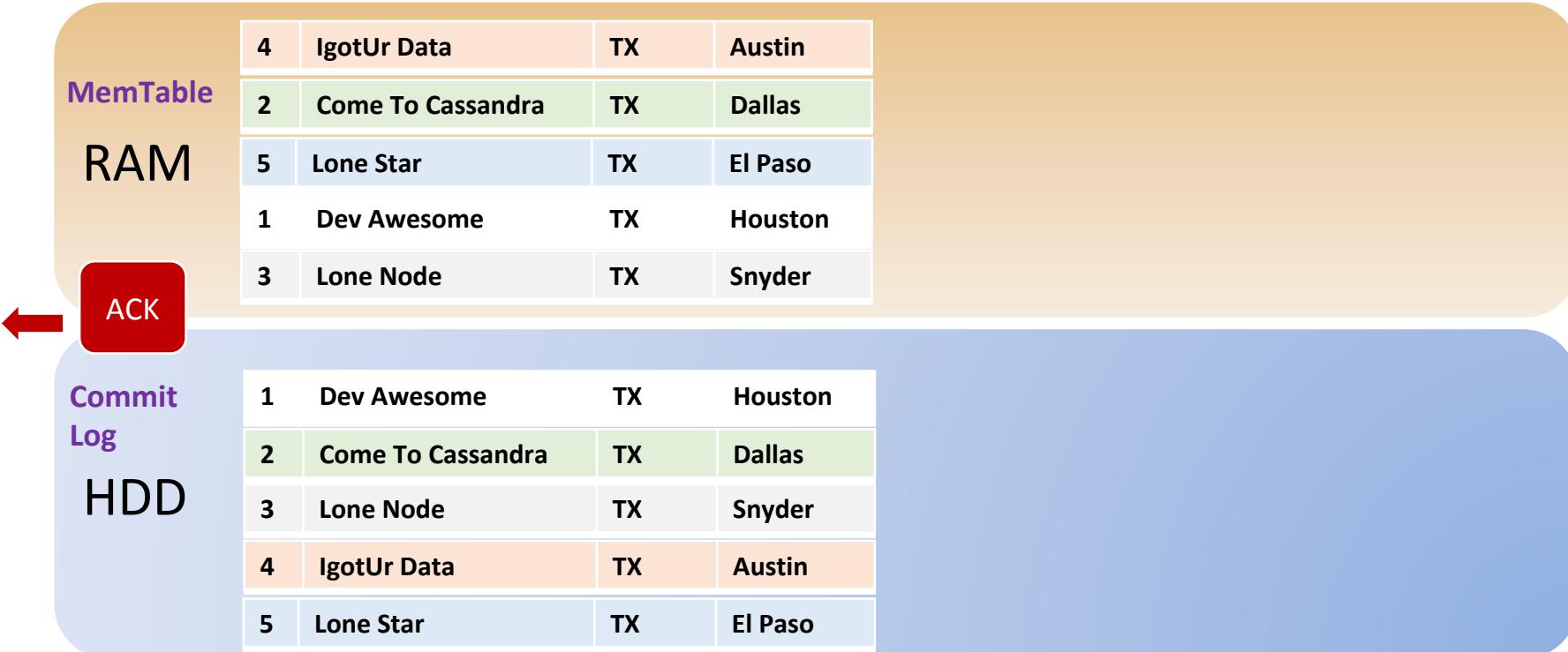
5	Lone Star	TX	El Paso
---	-----------	----	---------

Commit
Log

HDD

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
HDD

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

HDD

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

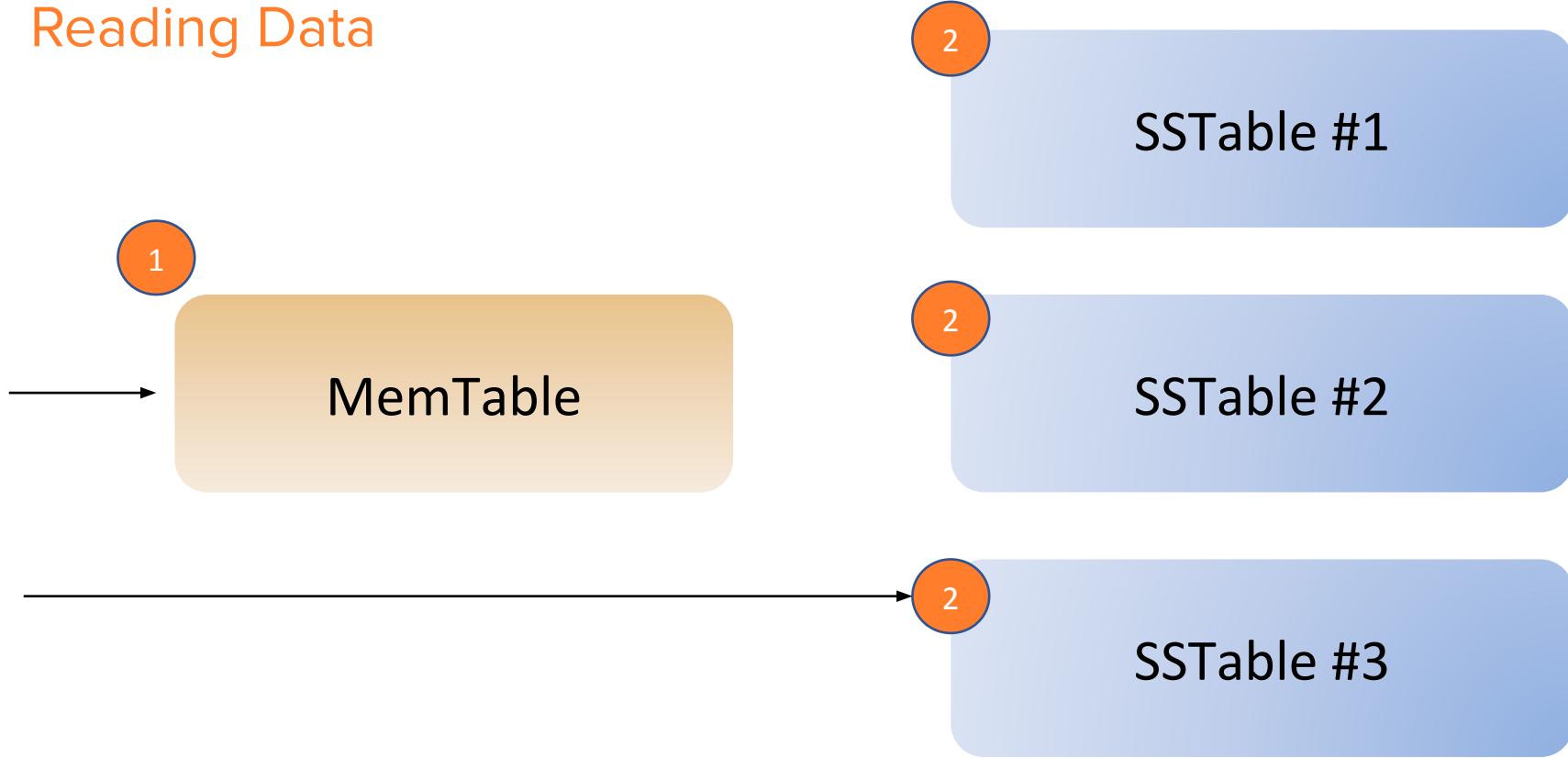
RAM

HDD

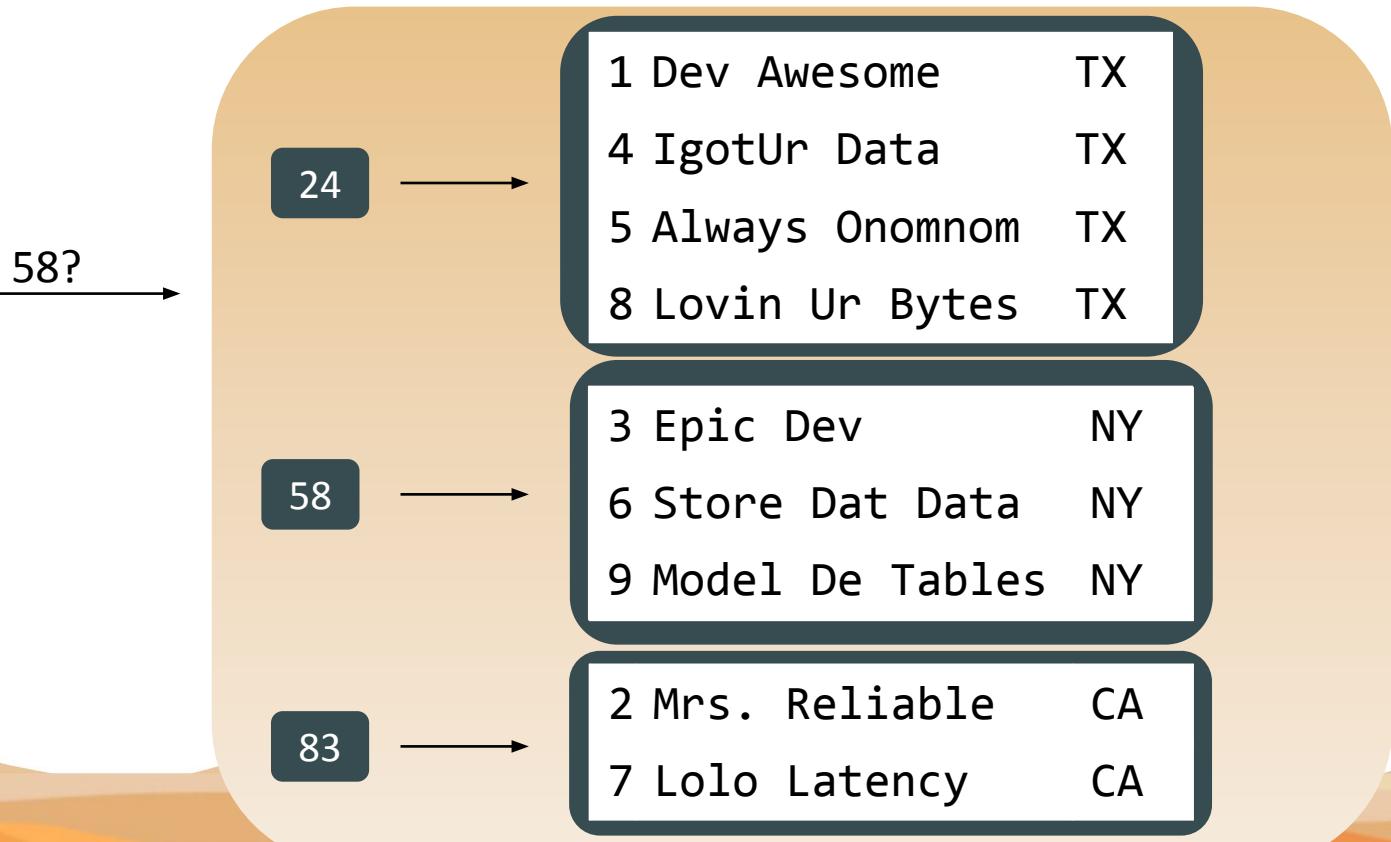
SSTABLE
(IMMUTABLE)

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

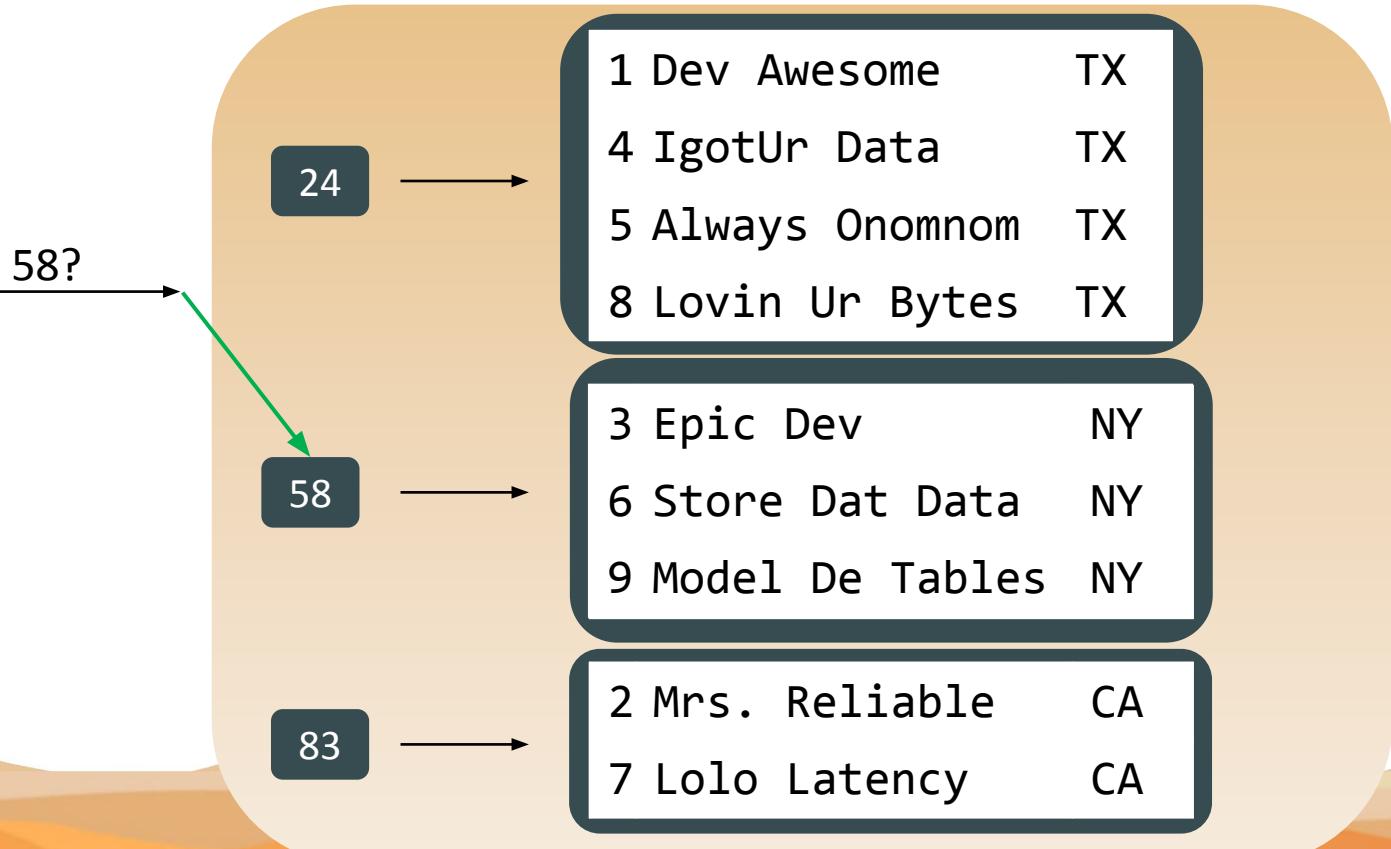
Reading Data



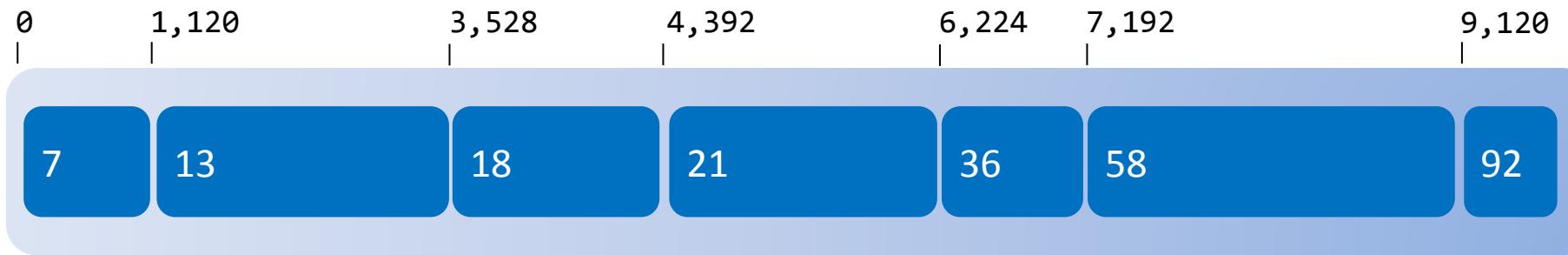
Reading a MemTable



Reading a MemTable

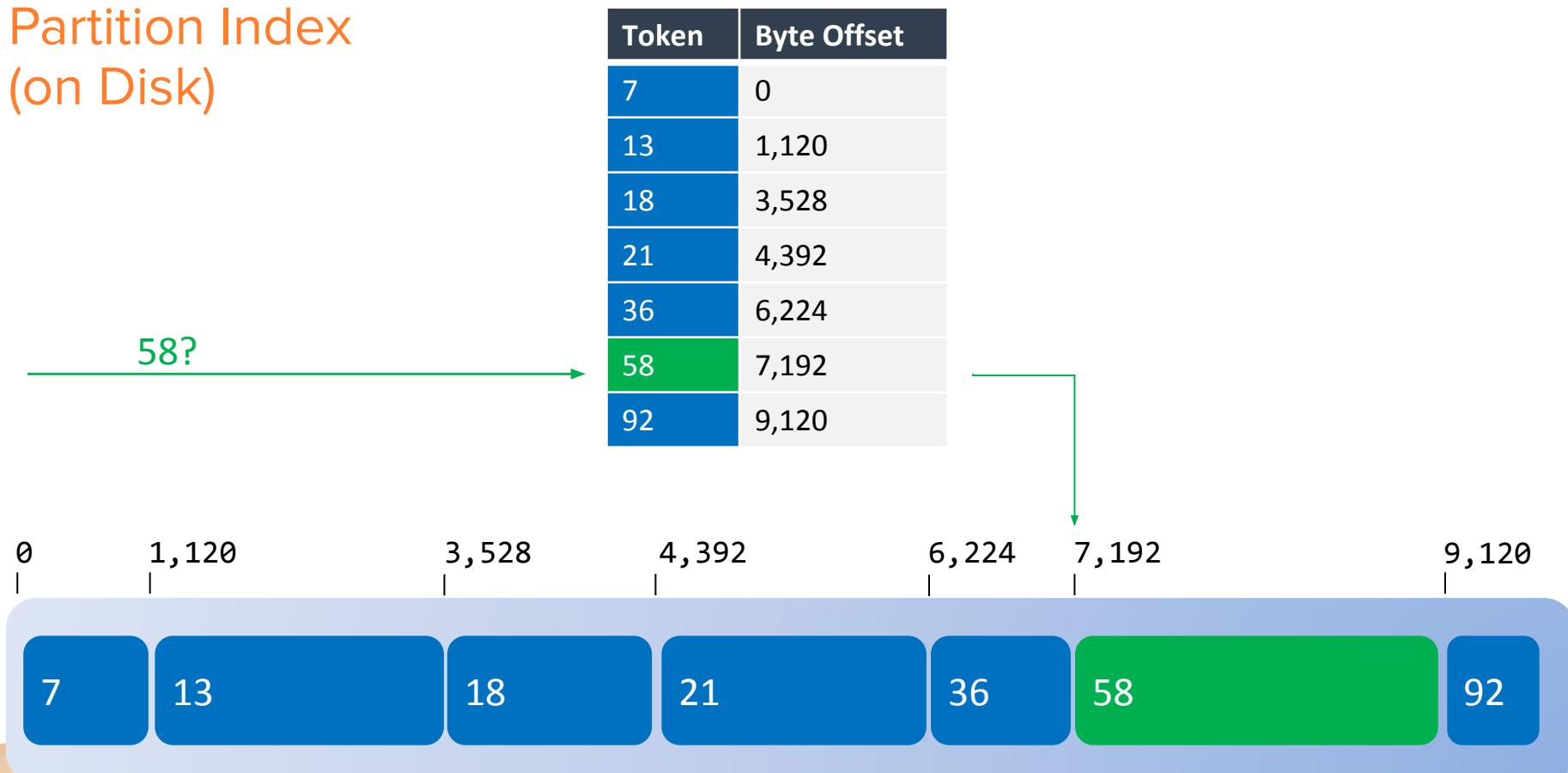


Reading a SSTable

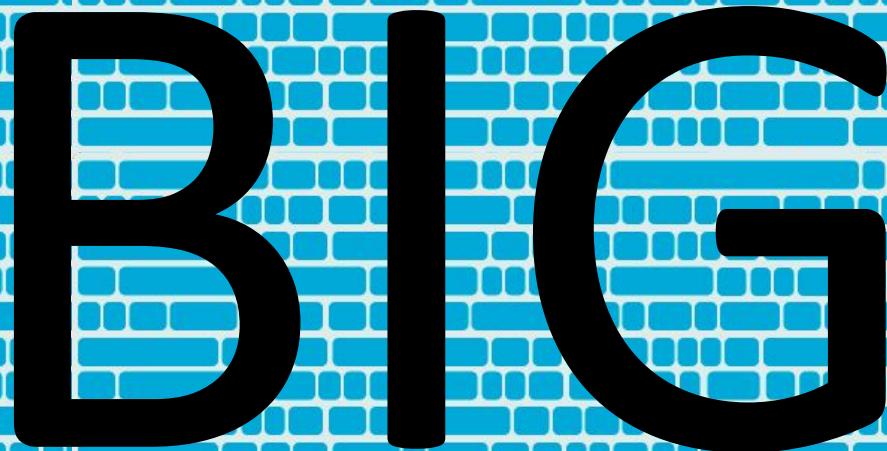


- ❖ A SSTable holds ordered partitions
- ❖ A partition can be split in multiple SSTables
- ❖ We can mark offset of each partition

Partition Index (on Disk)

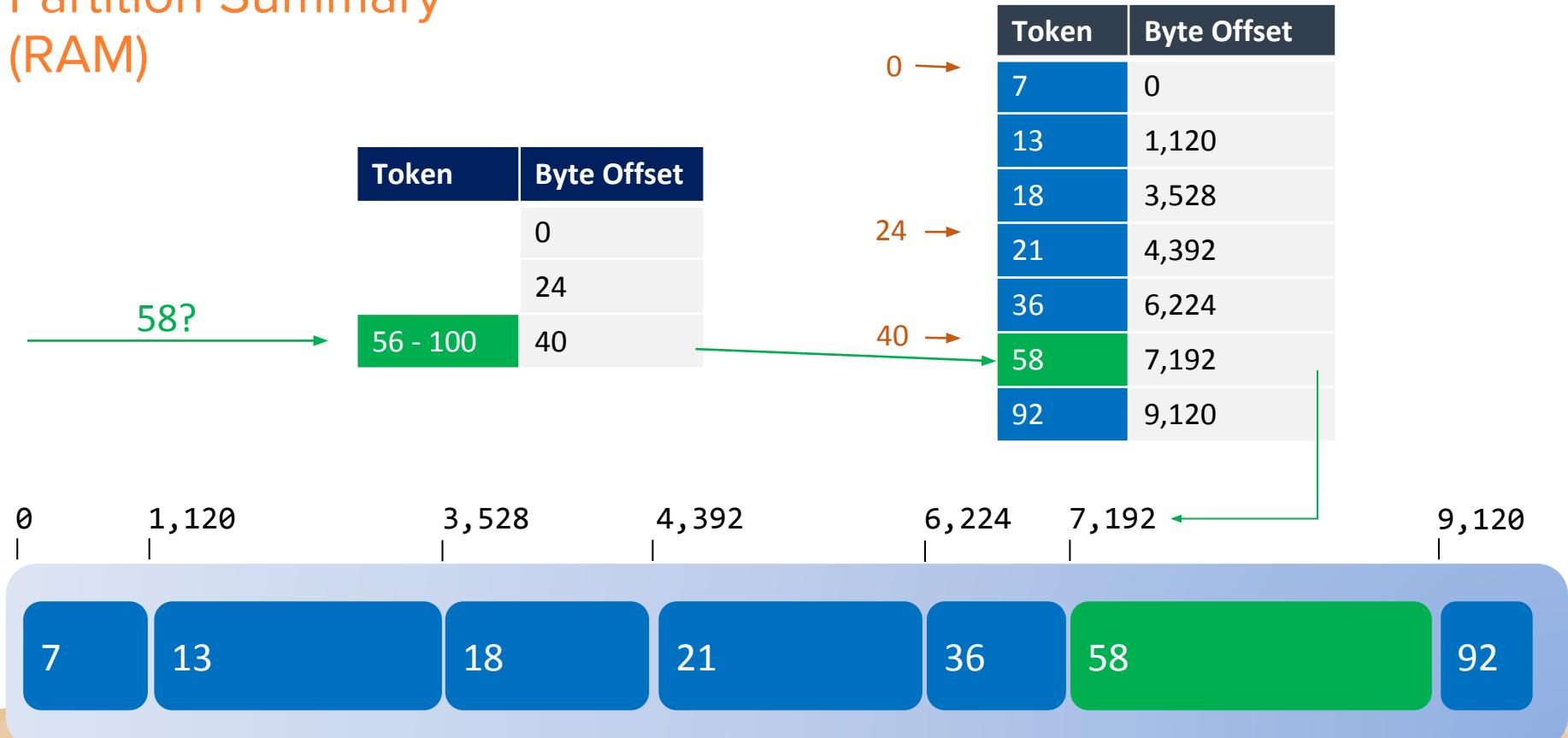


...but a SSTABLE is ...



BIG

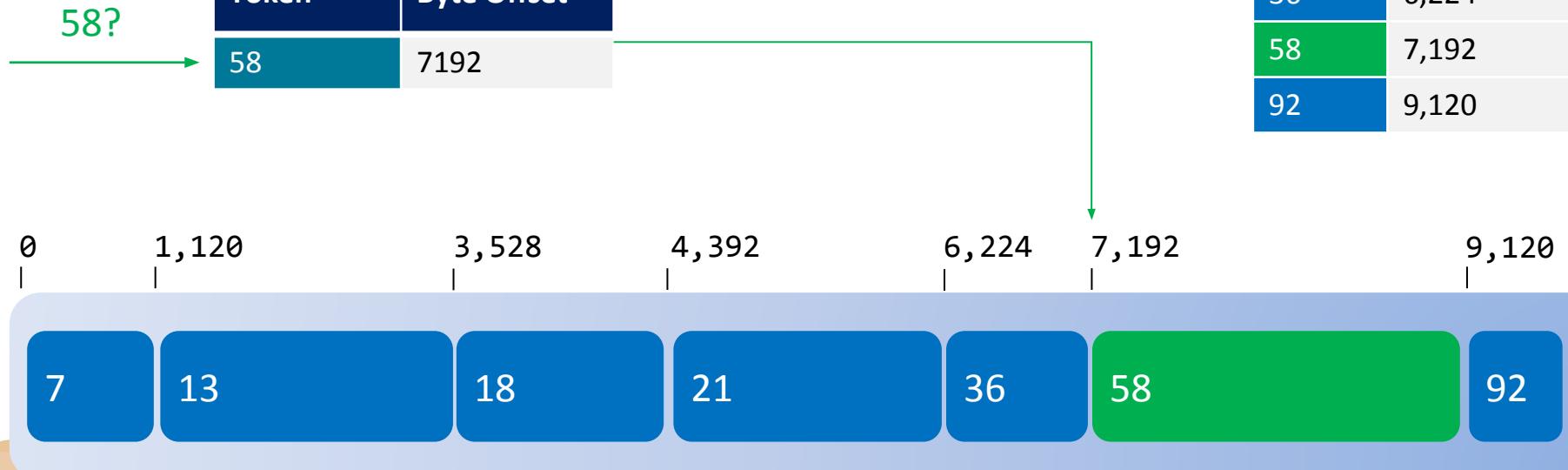
Partition Summary (RAM)



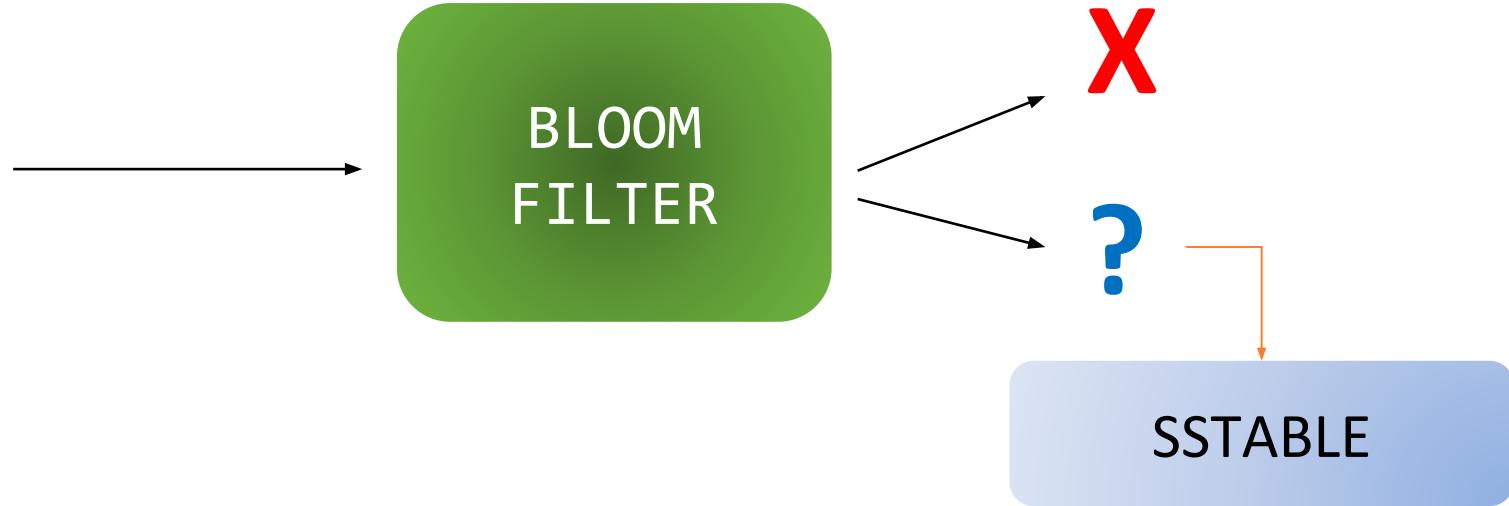
Key Cache (RAM)

Token	Byte Offset
0	
24	
40	

Token	Byte Offset
58	7192



Bloom Filter

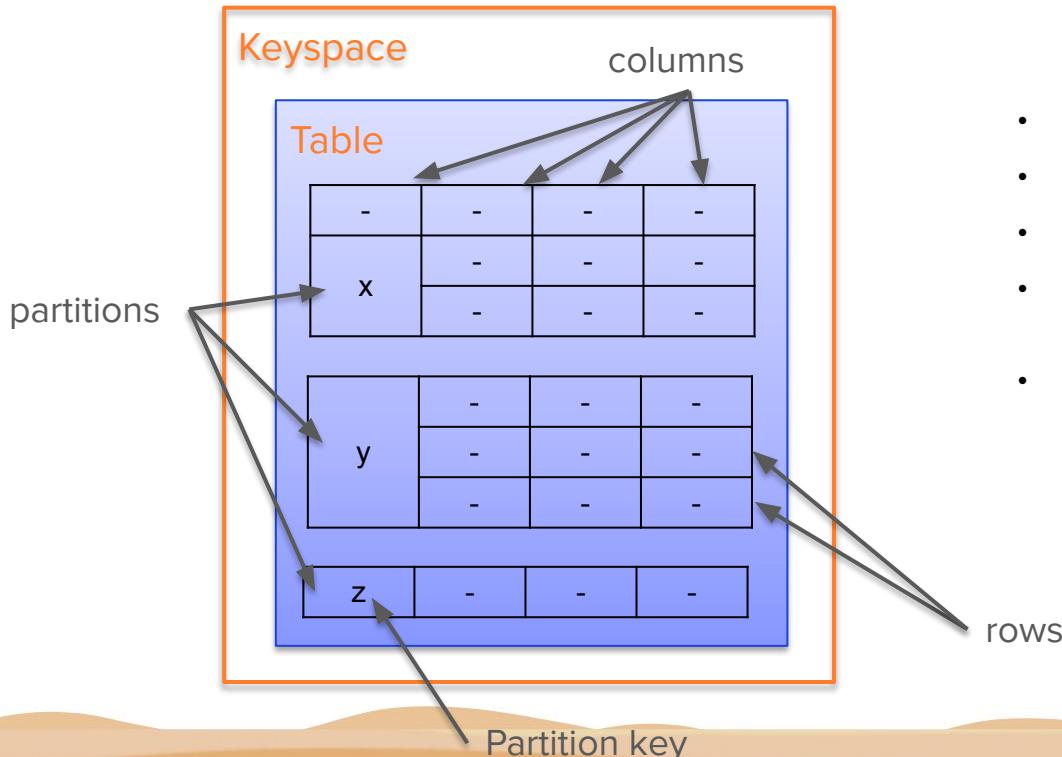




Developer Workshop Series

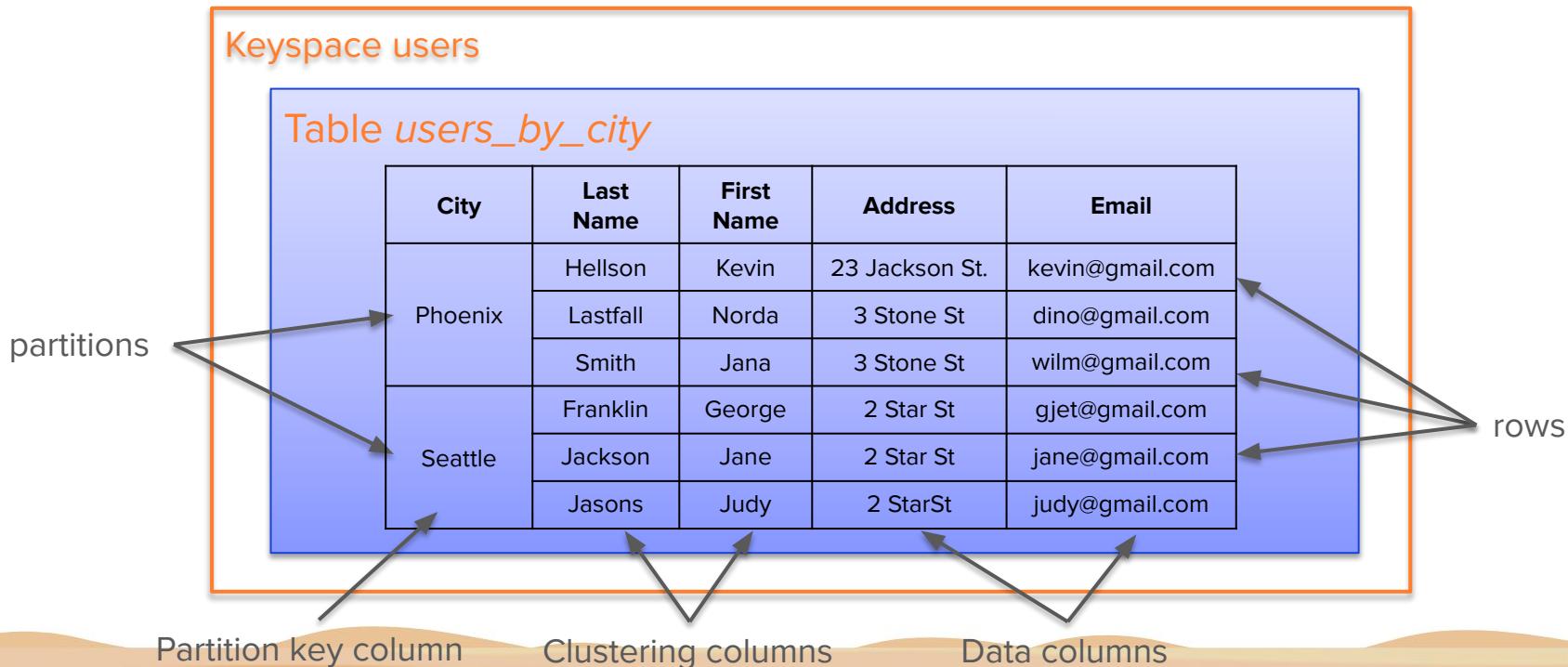
- Bootstrapping
- Apache Cassandra™ Why, What & When
- Read and Write path
- Uber High Level Data Modeling
- What's NEXT?

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

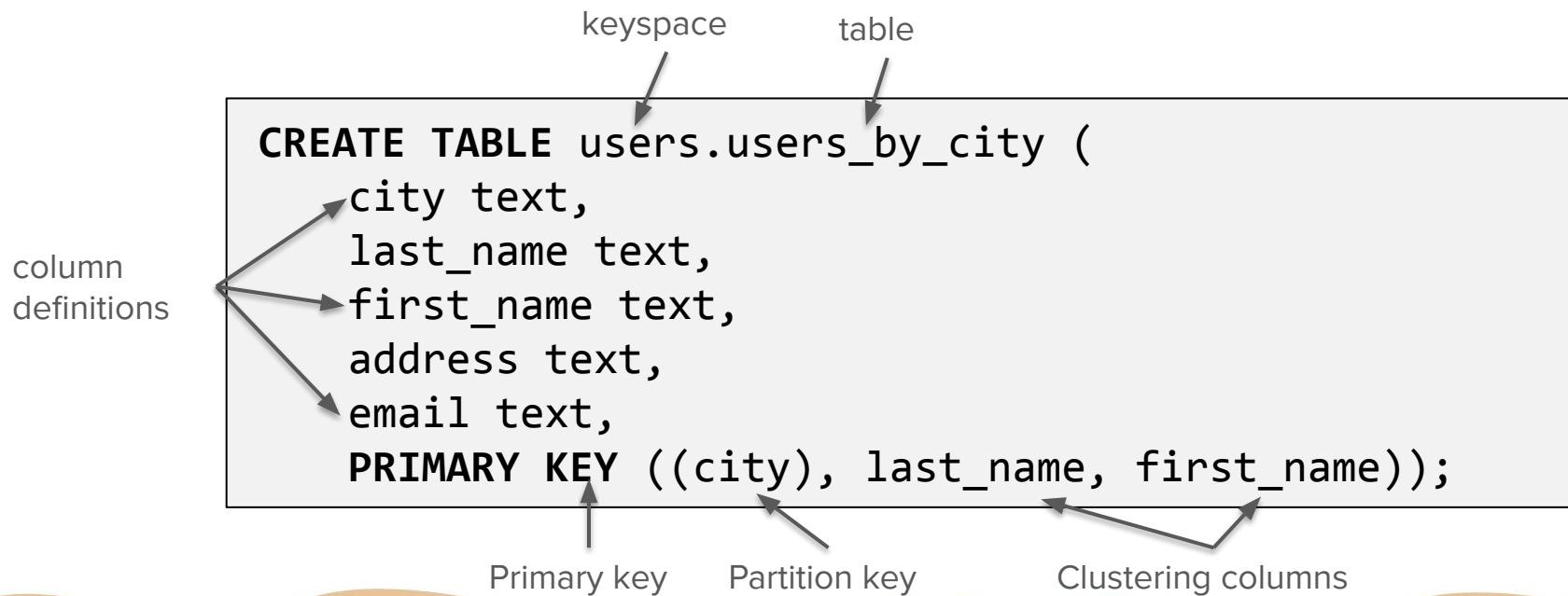
```
CREATE KEYSPACE users
  WITH REPLICATION = {
    'class' : 'NetworkTopologyStrategy',
    'datacenter1' : 3
};
```

keyspace

replication strategy

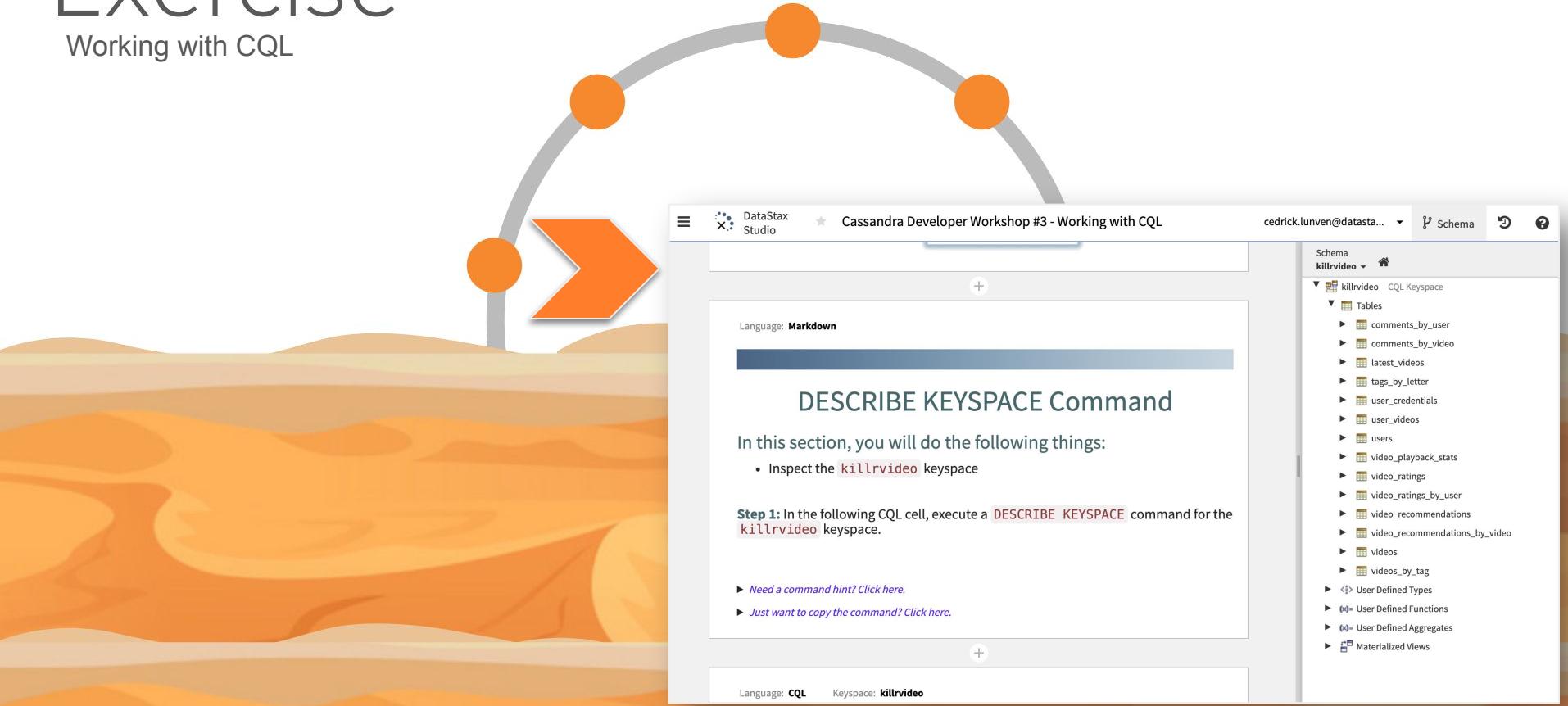
Replication factor by data center

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 displays a Markdown page titled "DESCRIBE KEYSPACE Command". The page content includes instructions for inspecting the "killrvideo" keyspace and executing a CQL command. The right sidebar shows the schema for the "killrvideo" keyspace, listing various tables and other database objects.

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.*

Language: **CQL** Keyspace: **killrvideo**

Schema
`killrvideo` CQL Keyspace

- 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
- User Defined Types
- User Defined Functions
- User Defined Aggregates
- 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?

Developer Resources

LEARN

Join academy.datastax.com

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

Free online courses - Cassandra certifications

ASK/SHARE

Join 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>

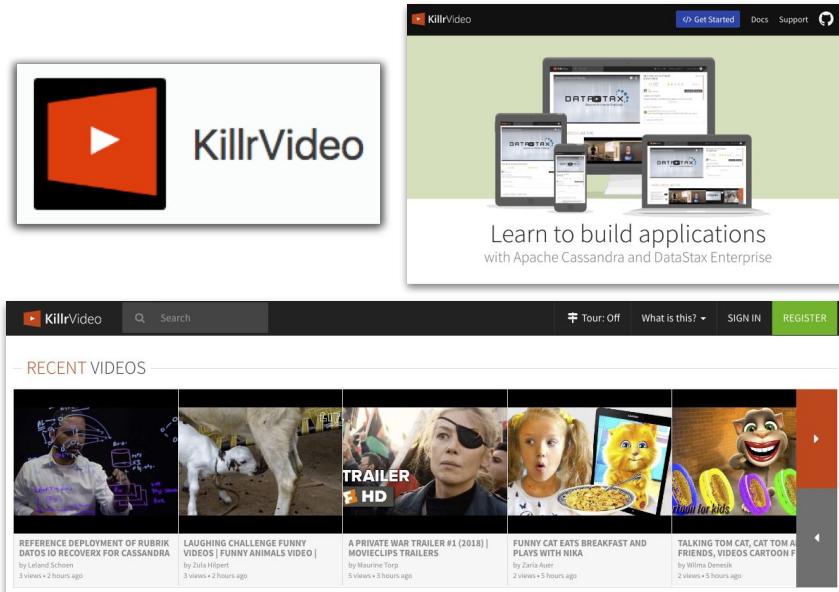
Training Courses at DataStax Academy

- 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!



Upcoming events

Date	Time	Content	Type
5/13	11am EDT	Cassandra Kubernetes Operator	WORKSHOP = HANDS-ON
5/18	12pm IST	Cassandra Developer Workshop (SAME)	WORKSHOP = HANDS-ON
5/20	12pm PT	Astra BetterBoz	WORKSHOP = HANDS-ON
5/27	12pm EDT	OSS Fallout	WORKSHOP = HANDS-ON
6/3	12pm EDT	NoSQLBench : Benchmark Your Data Models	WORKSHOP = HANDS-ON



Thank You



Week



Cassandra Workshop Series

Your path to becoming a Cassandra expert!



What we will
cover:



#CassandraWorkshopSeries







#CassandraWorkshopSeries



Exercise

