DataStax Hands-On Workshop

Giscard Venn
Negib Marhoul

1. March 2018

Agenda

1	Quick intro to DataStax and Team
2	Key Facts DataStax Enterprise
3	Hands-On Agenda today
4	DataStax Enterprise Multimodel Architecture

Design for failure and nothing will fail

DATASTAX

Apache Cassandra™ Architecture

Cluster layer

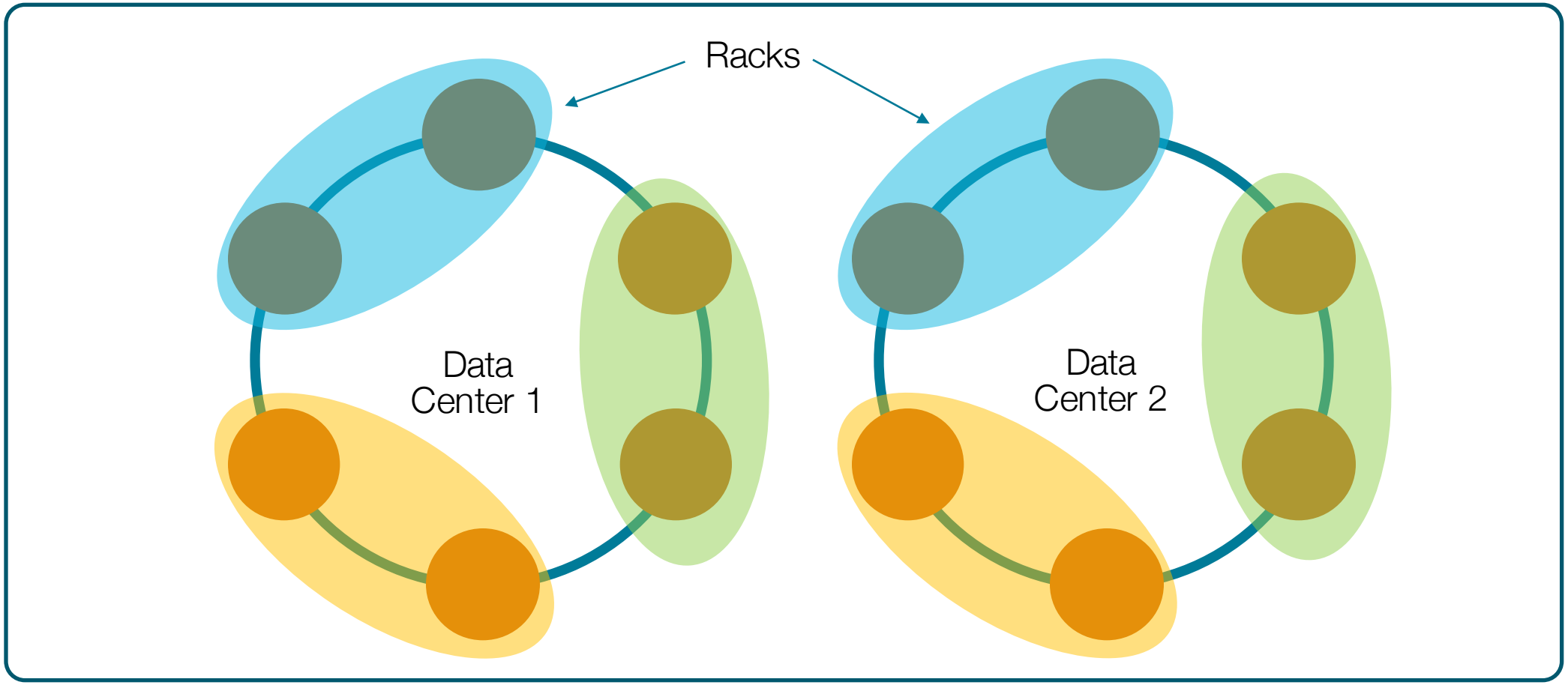
- Amazon DynamoDB paper
- masterless architecture

Data-store layer

- Google Big Table paper
- Columns / columns family

Topology

Cluster



Master-less Always-On, Scalable, Distributed

Continues Availability

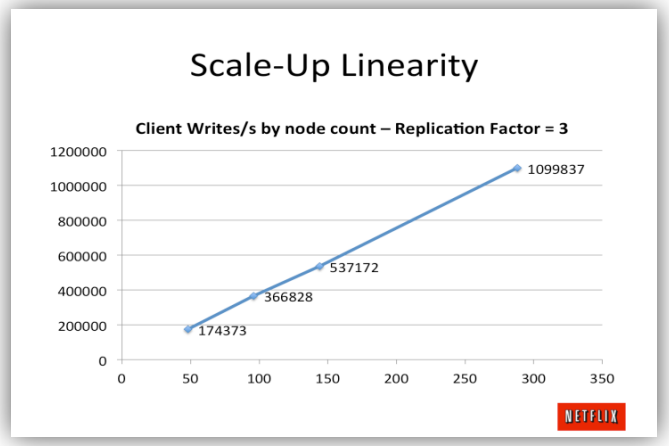
- No master, Master Less
- Topology discovery
- Client topology awareness

Linear Scalability

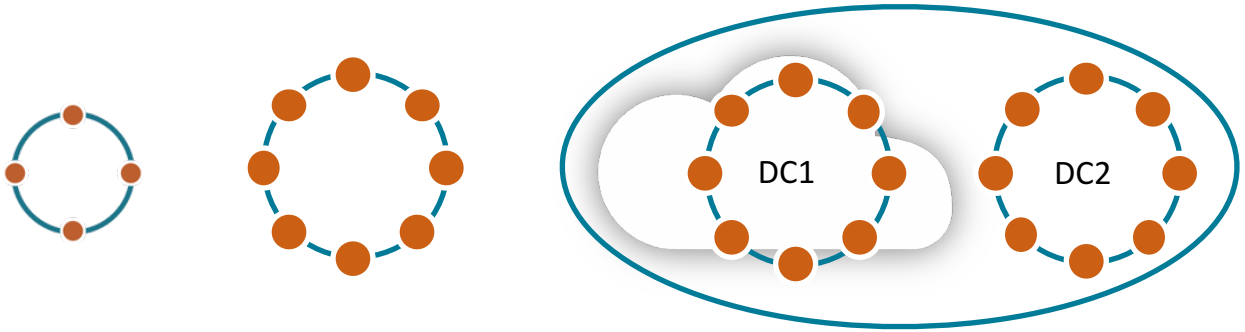
- Scale out
- tunable consistency
- Runs on Commodity hardware

Built-in data distribution

- Shared-Nothing Architecture
- coordination free
- Automatic data distribution



http://techblog.netflix.com/2011/11/benchmarking-cassandra-scalability-on.html

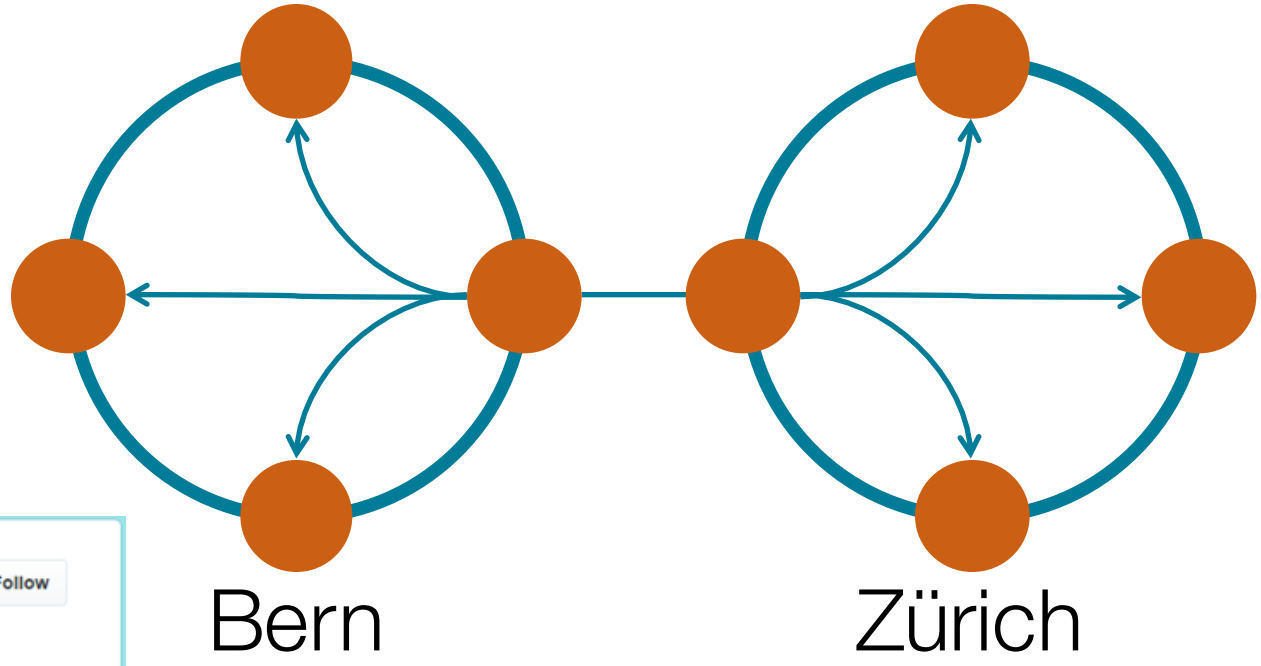


AlwaysOn, Linear Scalability On Premise, Cloud or Hybrid

DATASTAX

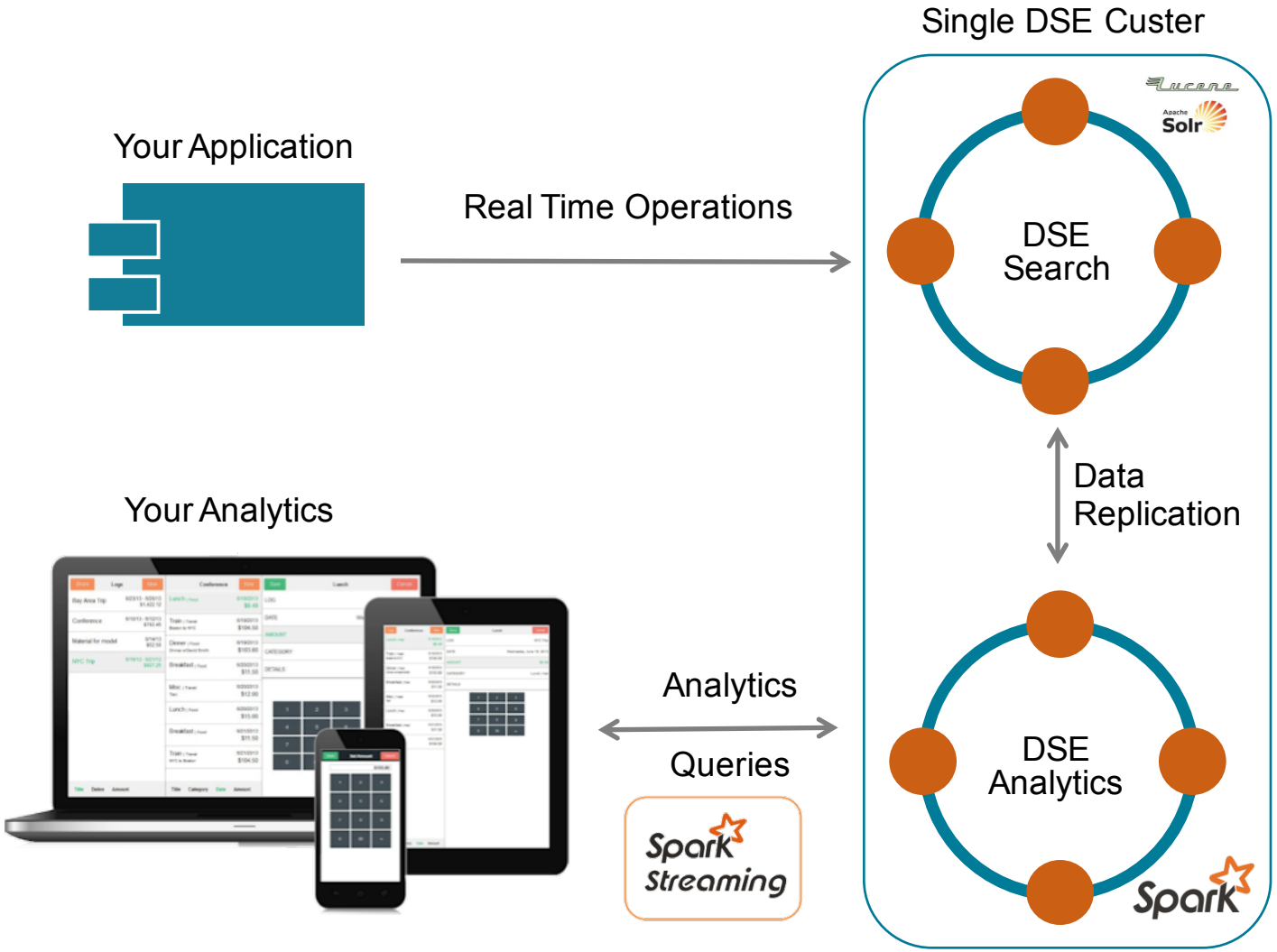
Distributed Cluster

- Cross datacenter synchronization
- On-Prem and on Cloud Environments
- Workload Segregation
- Client Driver is Topology Aware
- LWW Concept
- NTP needed



DATASTAX

DSE Reference Architecture



Streaming, ad-hoc, and batch

- High-performance
- Workload management
- SQL reporting

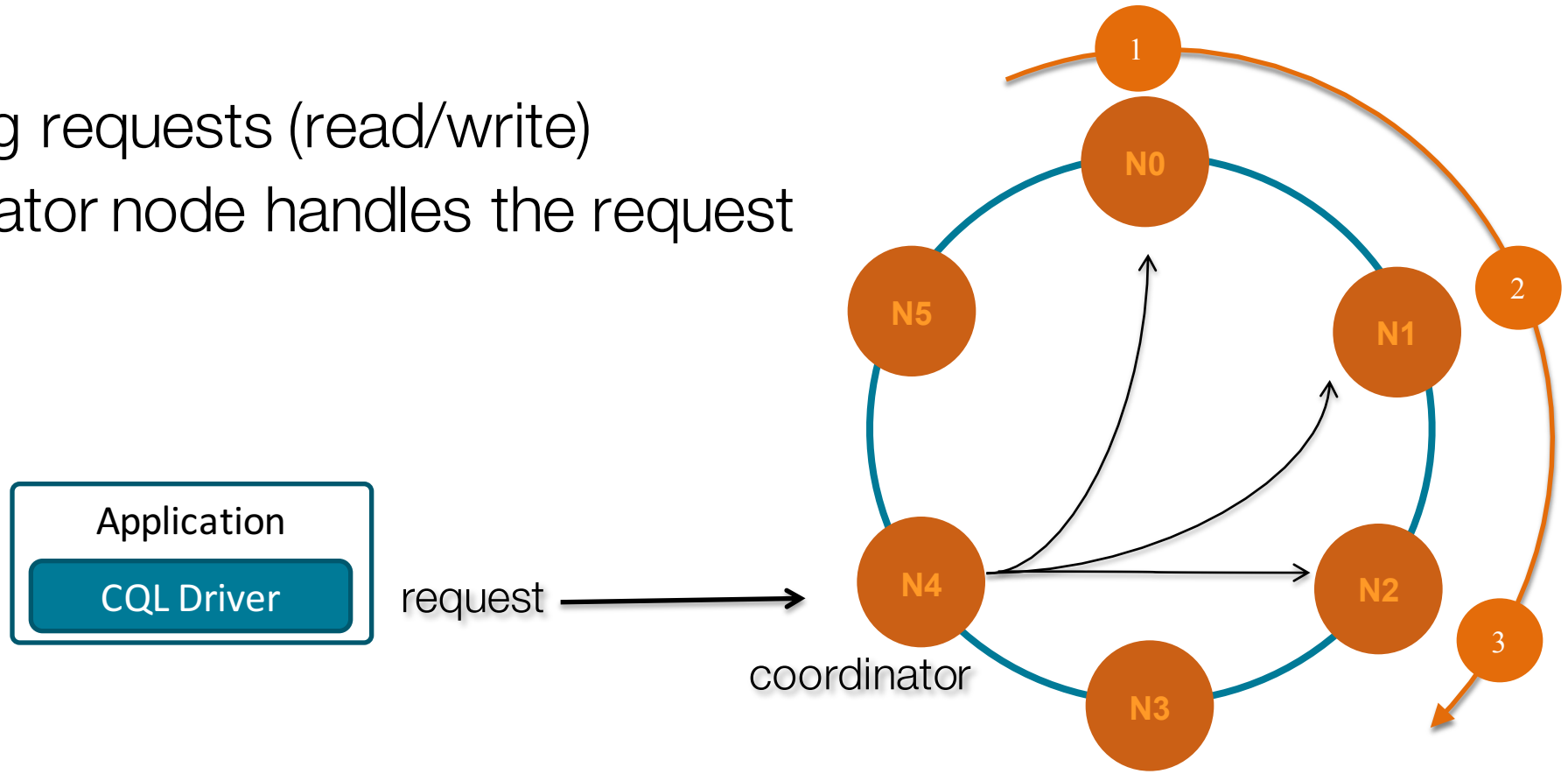
Compared to self-managed:

- No ETL
- True HA without Zookeeper

DATASTAX

Coordinator node

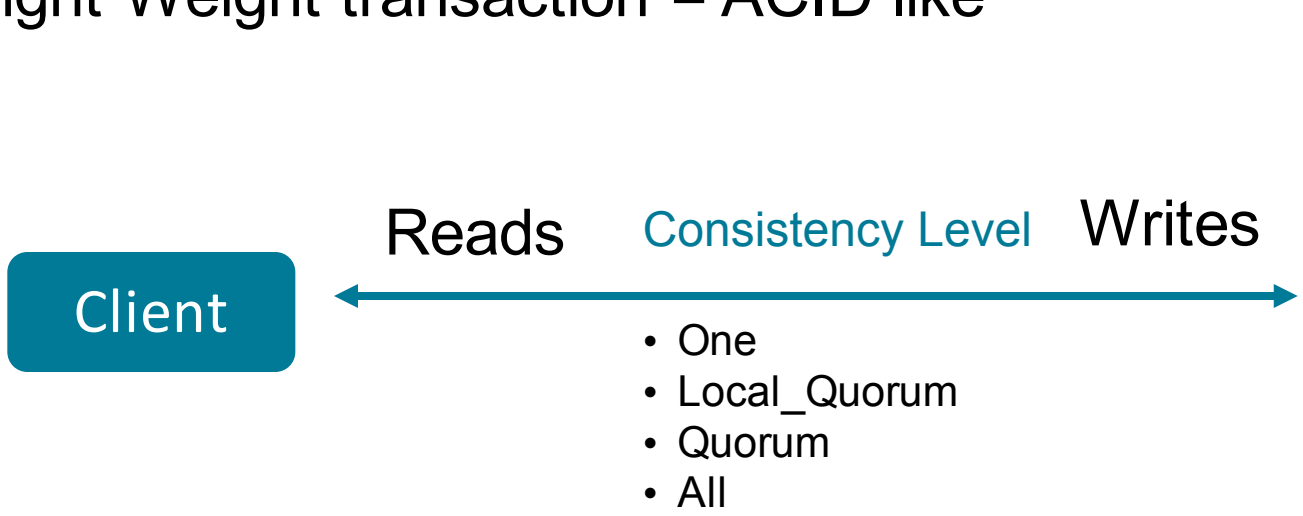
- Incoming requests (read/write)
- Coordinator node handles the request

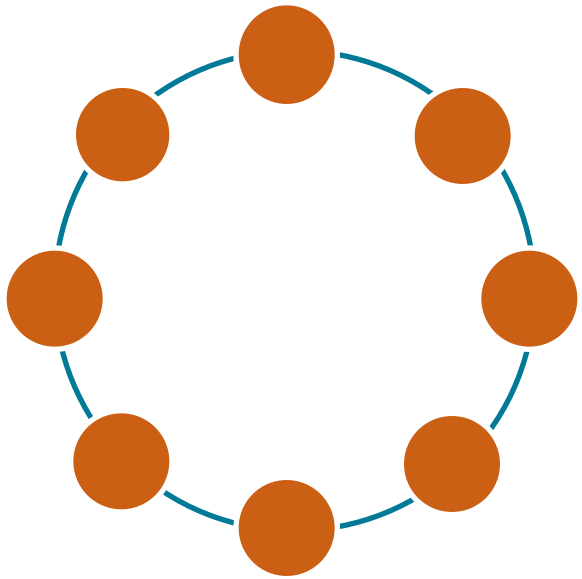


Every node can be coordinator → masterless

Tunable consistency

- Choose between strong and eventual consistency depending on the need
- Can be done on a per-operation basis, and for both reads and writes
- Handles multi-data center operations
- Light Weight transaction = ACID like





Anti-Entropy and Consistency

Write time

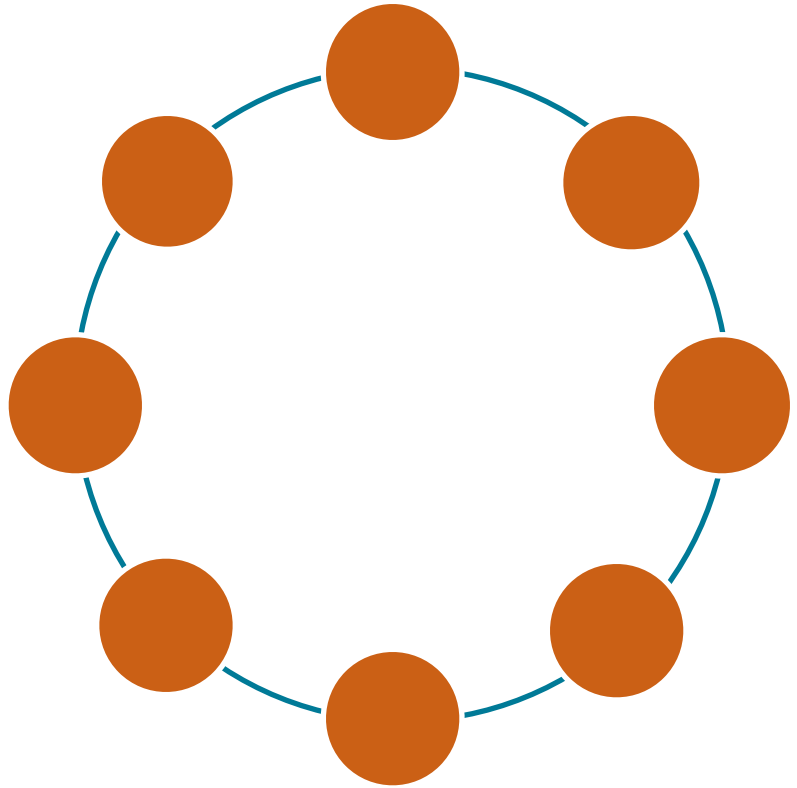
- Tunable Consistency
- Atomic batches
- Hinted handoff

Read Time

- Consistent reads
- Read Repair

Maintanance Time

- Node repair



Lab 1 : Accessing the cluster

DATASTAX

Uniform Data Distribution

Query Based Modelling

DATASTAX

Primary data model

Name
Value

Column

Partition	Name	Name	Name	…
	Value	Value	Value	…

Row

Partition	Wide Row Column		Wide Row Column		…		…		…		…	
	Name	Name	Name	Name	Name	Name	Name	Name	Name	Name	Name	Name
	Value	Value	Value	Value	Value	Value	Value	Value	Value	Value	Value	Value

Wide Row

- Row-oriented, column structure
- Table: similar to an RDBMS table but more flexible/dynamic
  - A row in a table is indexed by its key

DATASTAX

Modelling explicit partitioning

```
CREATE KEYSPACE messaging WITH replication = {'class':
'NetworkTopologyStrategy', 'DC1': '3'}
AND durable_writes = true;
```

Replication Factor

```
CREATE TABLE messaging.notifications(
    target_user text,
    notification_id timeuuid,
    notification_time timestamp,
    ...
    activity text,
    PRIMARY KEY (target_user, notification_time)
) WITH CLUSTERING ORDER BY (notification_id DESC)
```

Partition Key

Clustering Column

Query Optimized

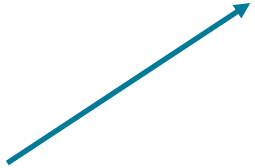
```
SELECT * FROM notification where target_user ='mike' limit 1;

SELECT * FROM notification where target_user ='mike' AND notification_time >= 2017-11-01 10:00;
```

DATASTAX

Primary Key – Unique Identifier

```
PRIMARY KEY ((account_number), transaction_time)
```



Partition Key

- Required to satisfy a queries' predicate(s)
- Ensures row uniqueness
- Defines the location of the partition in the cluster
  - Hashed to ensure even data distribution
- Can be composed of multiple columns
  - "Composite / Compound Key"

Clustering Key

- Sorts data within each partition
  Defaults to ascending order
- Can Be composed of multiple columns

Modelling explicit partitioning

target_user	notification_id	notification_time	activity
nick	5321998c	2017-11-01 10:00	tom liked
nick	ea1c5d35	2017-11-02 11:00	jake commented
nick	321998c	2017-11-03 09:00	mike created account
mike	e1bd2bcb	2017-11-01 07:00	tom created account

```
SELECT * FROM notification where
target_user ='nick' AND
target_user ='mike' AND
notification_time >= 2017-11-01 07:00;
```

Sorted by notification_time

nick	notification_time: 2017-11-01 10:00		notification_time: 2017-11-02 11:00		notification_time: 2017-11-03 09:00		....	notification_time: 2017-12-31 23:00	
	ntfcid: 5321998c	activity: tom liked	ntfcid: ea1c5d35	activity: jake commented	ntfcid: 5321998c	activity: mike created account			

Wide Row

Merged, Sorted and Stored Sequentially

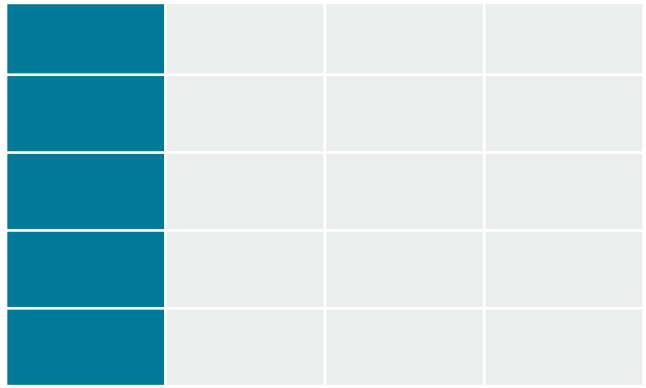
Tokens

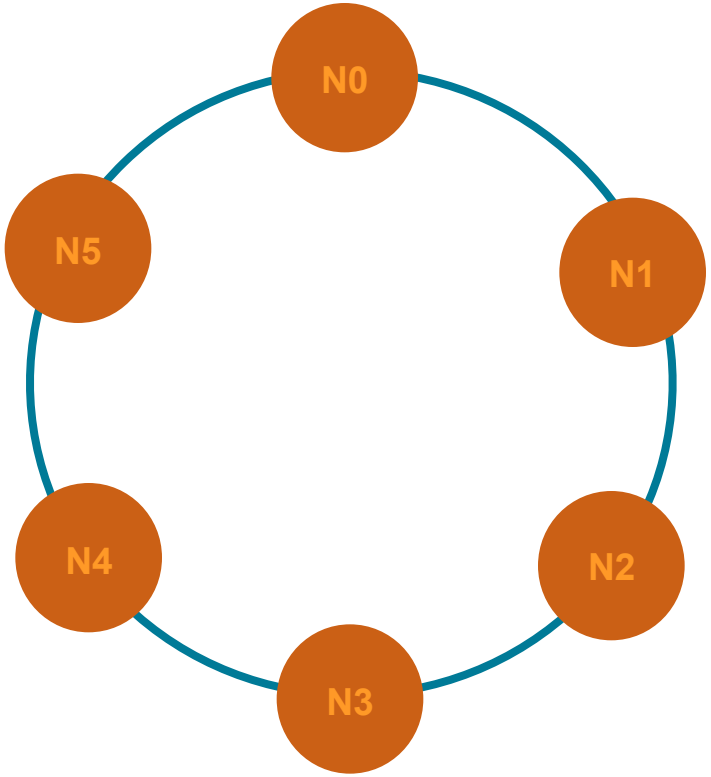
Token Range : - 2^{63} to 2^{63}

Data is partitioned after its partition key

A unique token is allocated to a partition

Token = *random* hash of #partition (murmer3)

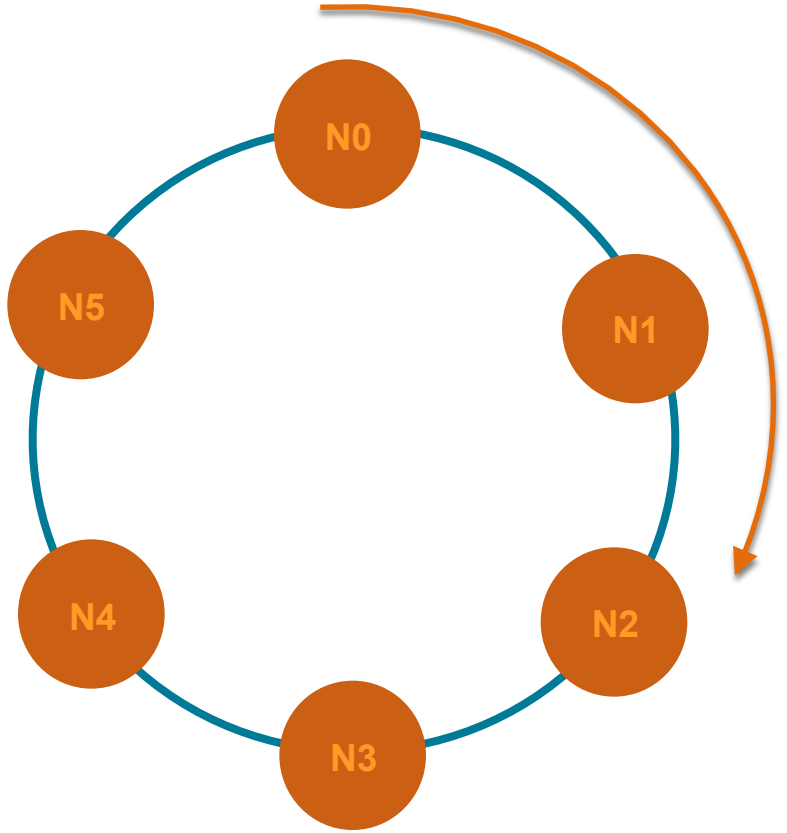




Data Distribution

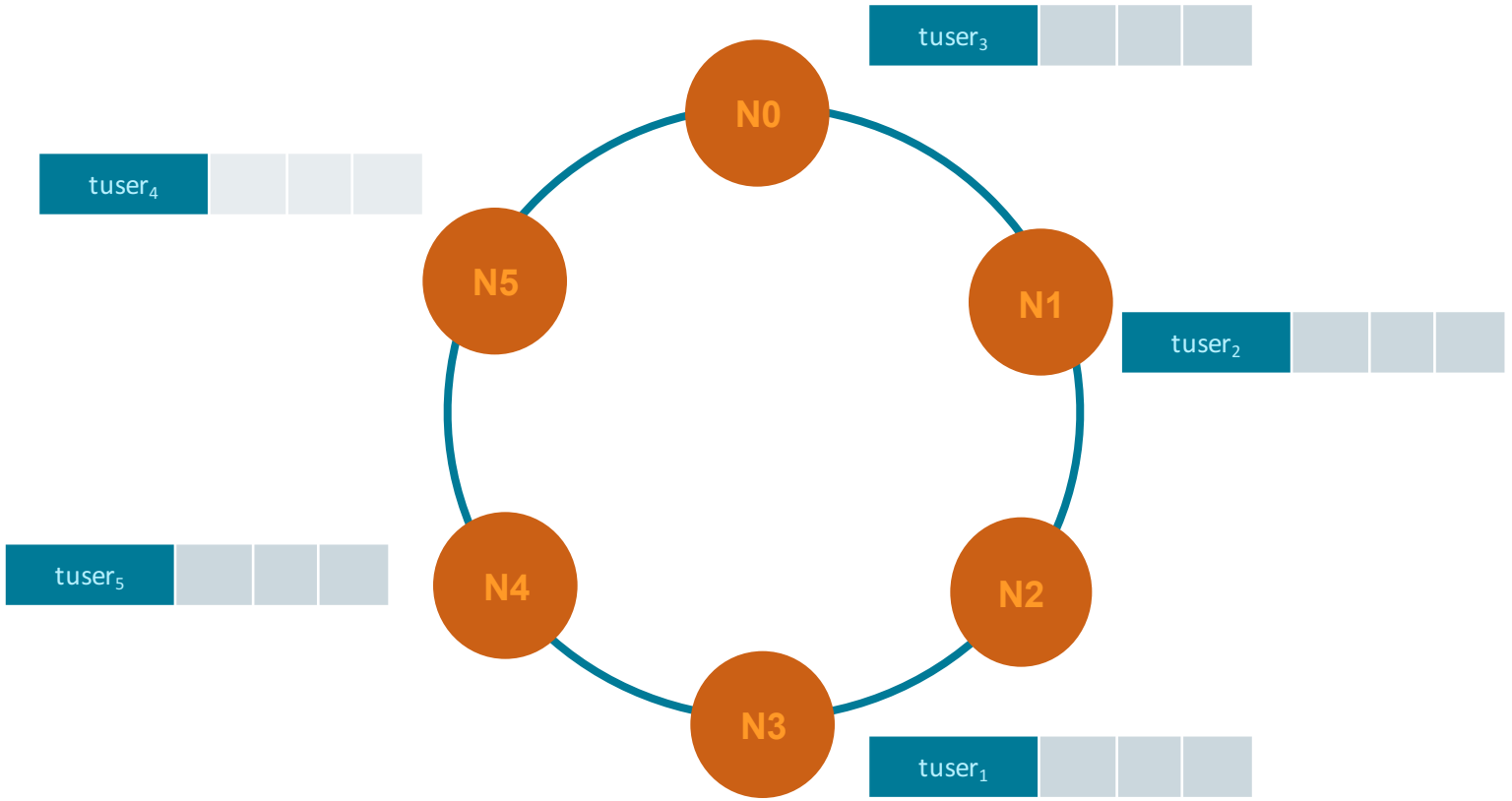
Token = hash of #partition → #node

Token1	target_user 1			
Token2	target_user 2			
Token3	target_user 3			
Token4	target_user 4			
Token4	target_user 5			



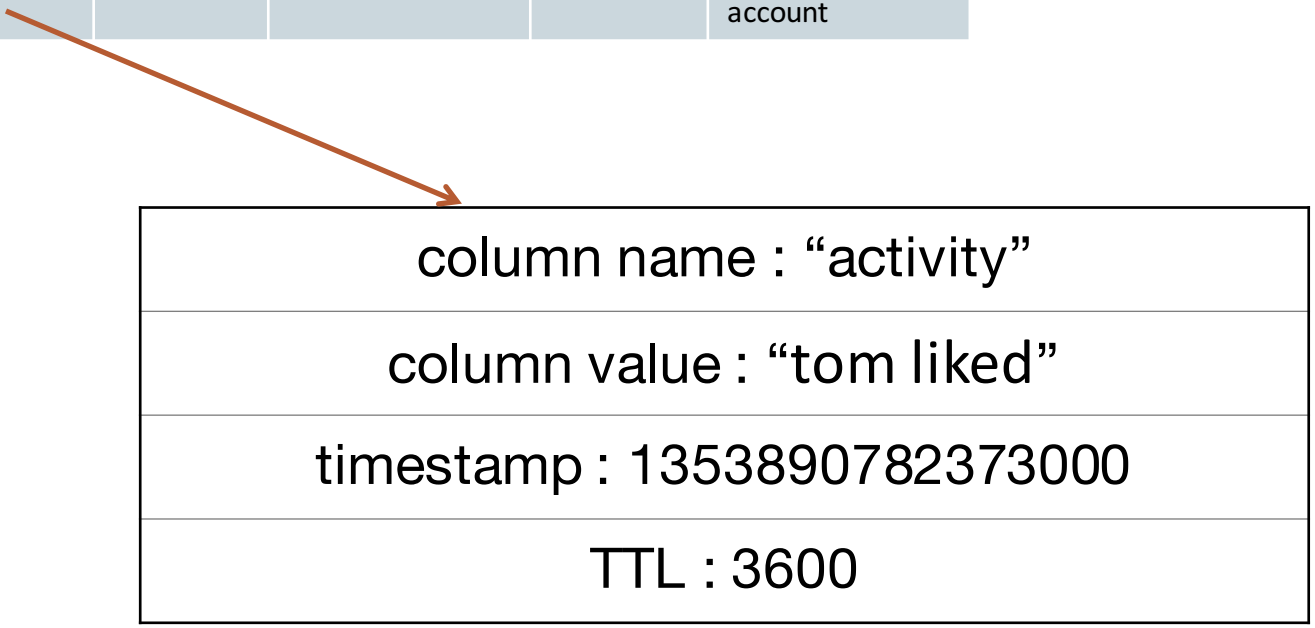
Data is evenly distributed and clock wise replicated

Automated Data Distribution Sharding



What's Stored With Each Column?

nick	notification_time: 2017-11-01 10:00		notification_time: 2017-11-02 11:00		notification_time: 2017-11-03 09:00	
	ntfcid: 5321998c	activity: tom liked	ntfcid: ea1c5d35	activity: jake commented	ntfcid: 5321998c	activity: mike created account



- Last Write Win, cross cluster clock sync, e.g NTP

Skinny vs. Wide rows

Compound Partition Key

```
PRIMARY KEY ((target_user,day), notification_time)
```

- Number columns per partition (2 billion max.)
- Faster operations and lower latency
- multiple gets per dataset if needed
- Equality select

```
SELECT * FROM notification where target_user ='mike' AND day IN (1,2,3);
```

Multiple Clustering Columns

```
PRIMARY KEY ((target_user), day, notification_time)
```

- Simulates 1-N relation ship
- wide rows
- Range selects

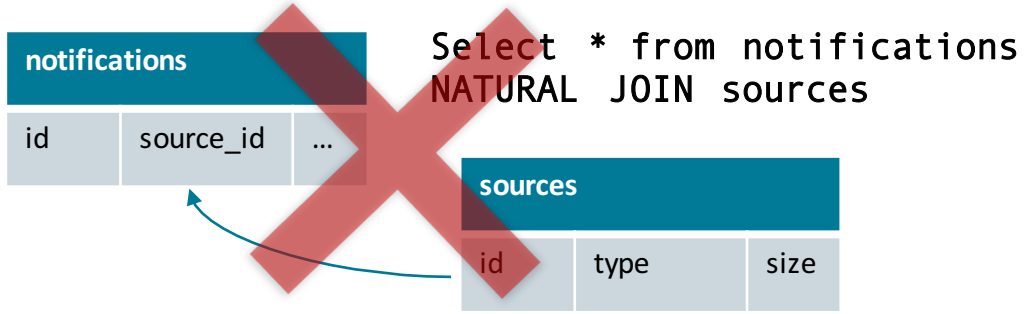
```
SELECT * FROM notification where target_user ='mike' AND day IN (1,2,3);
```

Basic Approach to Data Modeling

1. What queries are needed in your app?
2. What are your natural unique keys?
3. Is there ordering of the data needed to serve each query?
4. What are the groupings (1:M, M:M) in the data?
5. What filtering will your queries need?
6. Can events be stored in chronological order?
7. Does the data expire? Do large chunks of data expire together?

Alternatives to joins

- Collections
- Nested frozen Collections
- User Defined Types
- Nested Collections with UDTs
- JSON notation



Collections
<Values>

User Defined Types			
Name	Name	Name	Name
Value	Value	Value	Value

Partition	Name	Name	Collection	User Defined Types		
	Value	Value	<Values>	Name	Name	Collections
				Value	Value	<Values>

Row

DATASTAX

Alternatives to joins

```
CREATE TYPE source_type ( encoding text, size int, location text);

CREATE TABLE notifications(
    target_user text,
    notification_id timeuuid,
    notification_time timestamp,
    source map <text, frozen <source_type>>,
    activity text,
    PRIMARY KEY (target_user, notification_time)
) WITH CLUSTERING ORDER BY (notification_time DESC)


INSERT INTO notifications JSON '{
    "target_user ": “nick",
    "notification_id ": "5321998c",
    "notification_time ": “2017-11-01",
    “source": {
        “profile_pic": {
            “encoding": “jpeg",
            “size": 15,
            “location": “/"}}}';
```

DSE Performance Basics

Can DSE be both bigger and faster? Yes it can.

More Throughput? More Data?
Use more nodes (scale out)
Do not use too big nodes (~~scale up~~)
Know Cassandra ops best practices Use OpsCenter to monitor, alert, repair

Faster Operations? Predictable Latency?
Check your data model and queries
Use asynchronous queries
Use prepared statements
Compaction tuning or *maybe* strategy

Lab 2 : DSE Core and Operations

Introducing DataStax Managed Cloud

A Fully Managed, Secure Architecture

- DSE on AWS for production workloads
- 24x7x365 coverage, lights-out management – by the engineering and support experts at managing to scale
- Optimized for your apps
- Architecture advisory services, guidance, and best practices

Onboarding

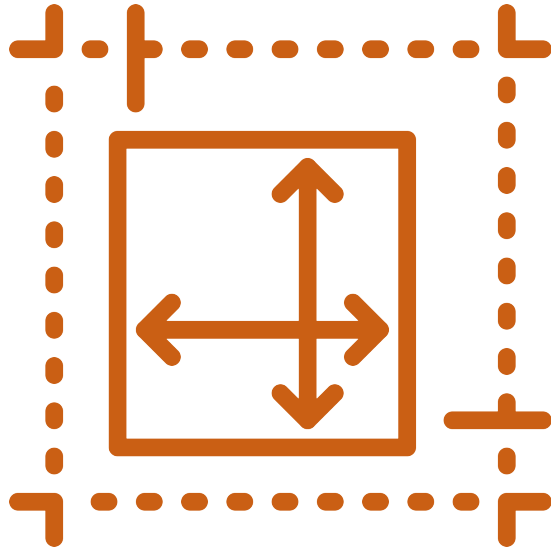
Ramping you up for take-off

- DataStax Solution Architect
  - Orients you to the cloud console and platform
  - Reviews your apps for platform, data model, and access patterns
- Cluster
  - Tested and sized for right instances
  - Production provisioned
  - Alerting configured

Provisioning

Custom fit

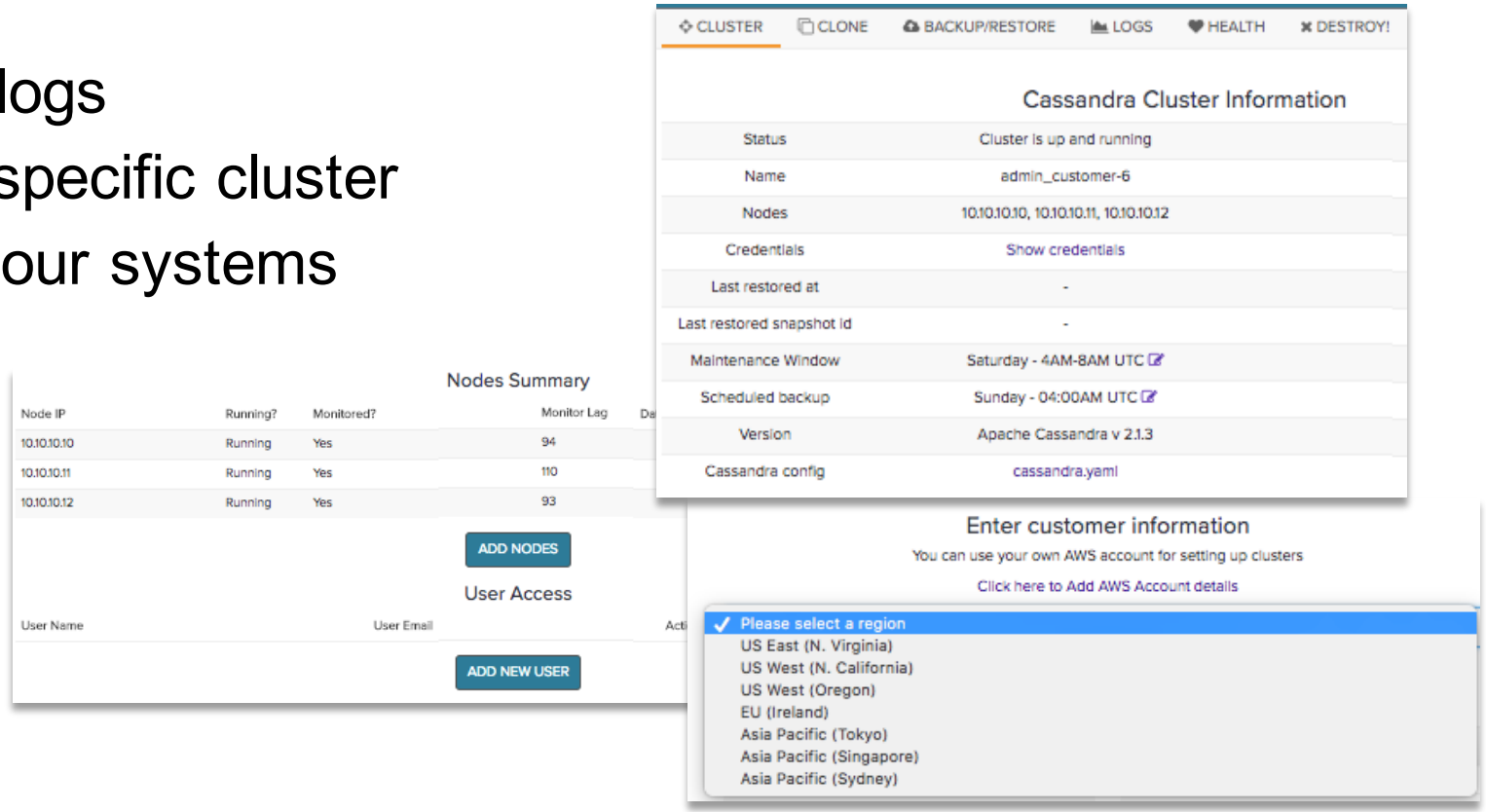
- Provisioning into your VPC
  - Environment setup, installation, and configuration
- Production
  - Best-practices blueprints
  - Configured for optimal performance
  - Tuned to your apps, lifecycle-managed
- Non-production (optional)
  - Ad-hoc, provisioning by end-user and lifecycle management via console or API



Cloud Console

Your “home base” to view your cluster

- View cluster metrics and logs
- Easily submit a ticket on specific cluster
- Console integrates with your systems via RESTful API

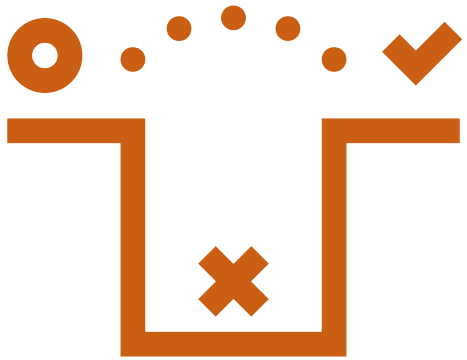


DATASTAX

Proactive intervention/alerting

Finding problems – before they exist

- Automated alerts against exceeded thresholds
  - Based on years of expertise
- Avoiding issues through
  - Automated fixes
  - Engineer action
- Fault-tolerance at multiple levels
  - Intervention still required for recovery – and handled for you
- Bursting included



Backups & restores

Restore with confidence

- Protect against
  - Data loss due to application bugs
  - Operator error
- Backups moved to cloud object storage for easy recovery
- Scheduled restore exercises – and restore on request

Upgrades support/service requests

Easy upgrades

- Aggressive bug fixes, if needed
  - Leveraging DataStax engineering for fast time to resolution
- The latest and greatest features
  – that fit your schedule

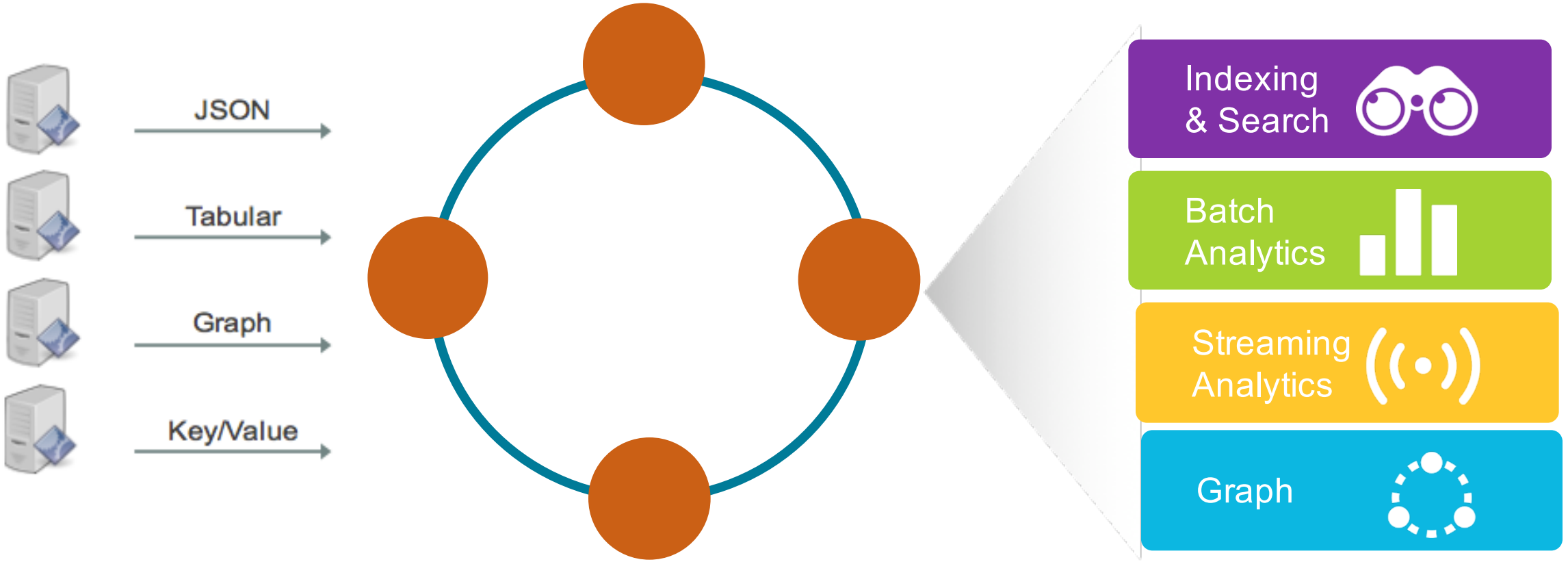
DATASTAX

Information Search

Query by non-PK

DATASTAX

Integrated Multi-Model Platform



DATASTAX

Information Search

Multi-criteria WHERE Constraints

- WHERE constraints with multiple columns
- No extra tables needed

Full Text Search

- **Wildcards** ? *, like or Lemmatisation
- **Faceting**, Slice and Dice

Live Indexing

- Real Time Search, High Index throughput

Geospatial queries

- Queries with coordinates and distances search

https://docs.datastax.com/en/dse/5.1/dse-dev/datastax_enterprise/search/tutorialsDemosTOC.html

https://github.com/phact/docker-silk-dse

DATASTAX

Query by non-PK

```
CREATE Search INDEX on notifications WITH COLUMNS activity;

SELECT * FROM notifications WHERE solr_query = 'activity:creat*';
```

Examples from the documentation

Geo-Spatial Search

```
SELECT * FROM test WHERE solr_query=
'{"q":"*:*", "fq":"point:\"IsWithin(BUFFER(POINT(4.0 49.0), 10.0))\""}';
```

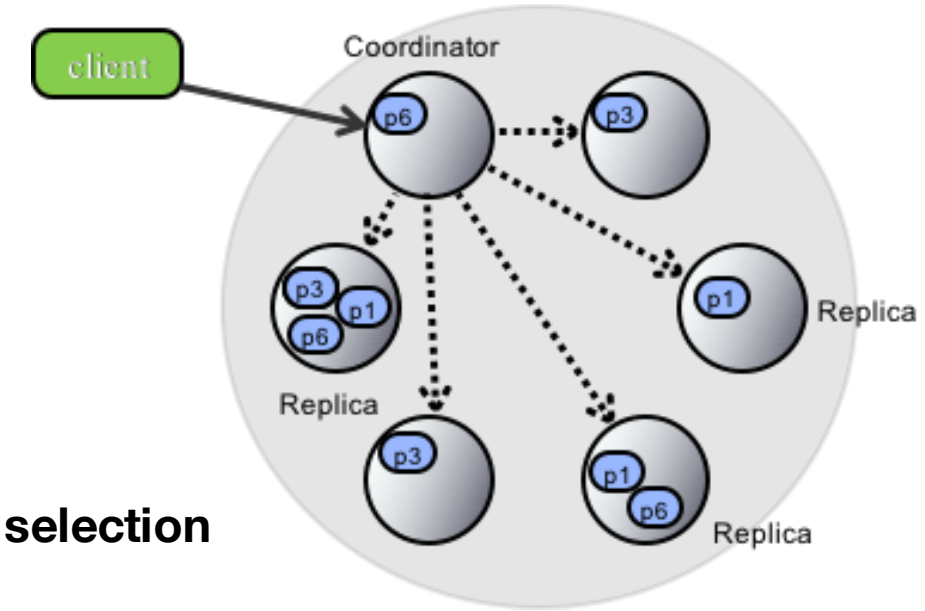
Range Queries (with open bounds possible)

```
SELECT * FROM taxi_trips WHERE solr_query =
'pickup_dropoff_range:[2017-02-01T12 TO 2017-02-02]';
```

Querying through CQL

```
SELECT activity FROM notifications WHERE solr_query = 'activity:creat*';

 activity
------------------------------
 mike created account
 tom created account
```



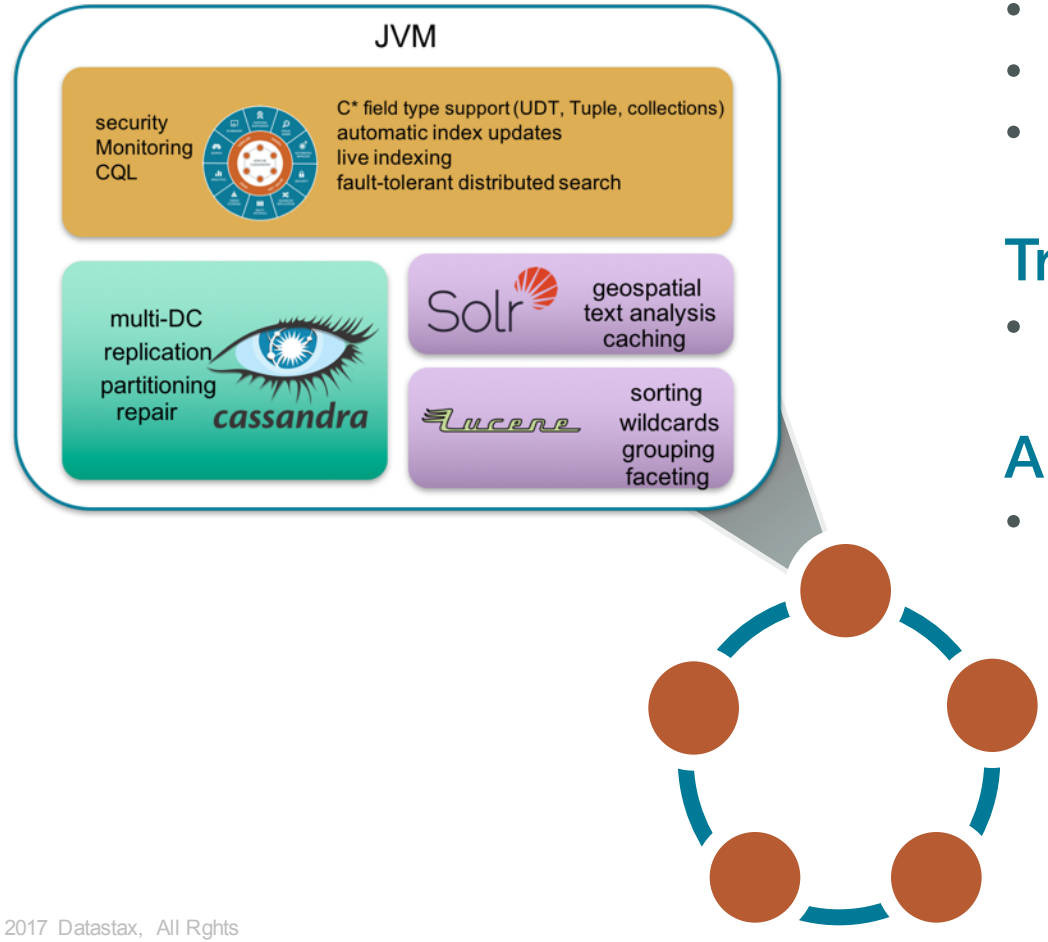
How many nodes to contact?

- We don't know the primary key
- Theory:
  - contact at least one replica for every token range

- Cassandra contacts all nodes
- **Our custom Solr SearchComponent does intelligent shard selection**

DATASTAX

DSE Search Architektur



Integrated in the the same JVM

- Data Locality (TokenRanges) and Shared Memory
- High Available, no master needed
- Index is sharded to all nodes

Transparency

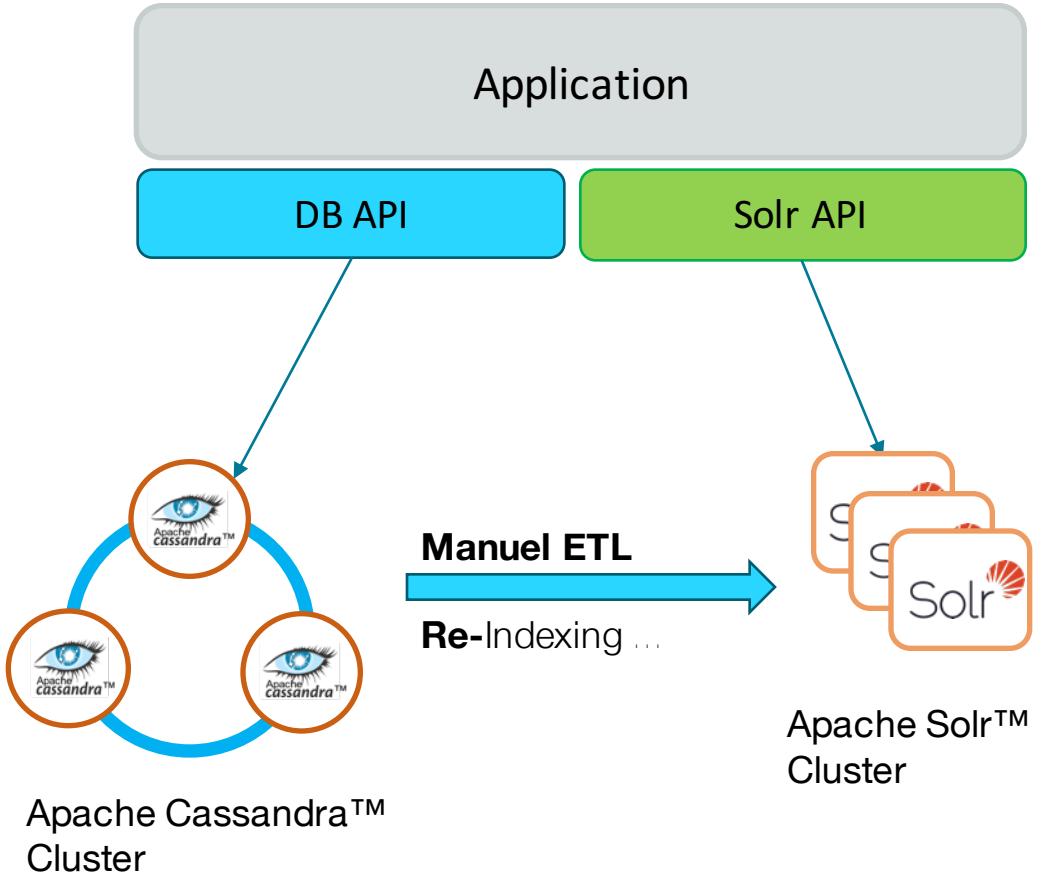
- Access via CQL or REST API

Automated indexing with INSERT / UPDATE

- No extra ETL Process needed

DATASTAX

The Open Source Way



Separated Search Cluster

- "Split Brain" risk, data inconsistency
- ETL to generate, update and re-create index

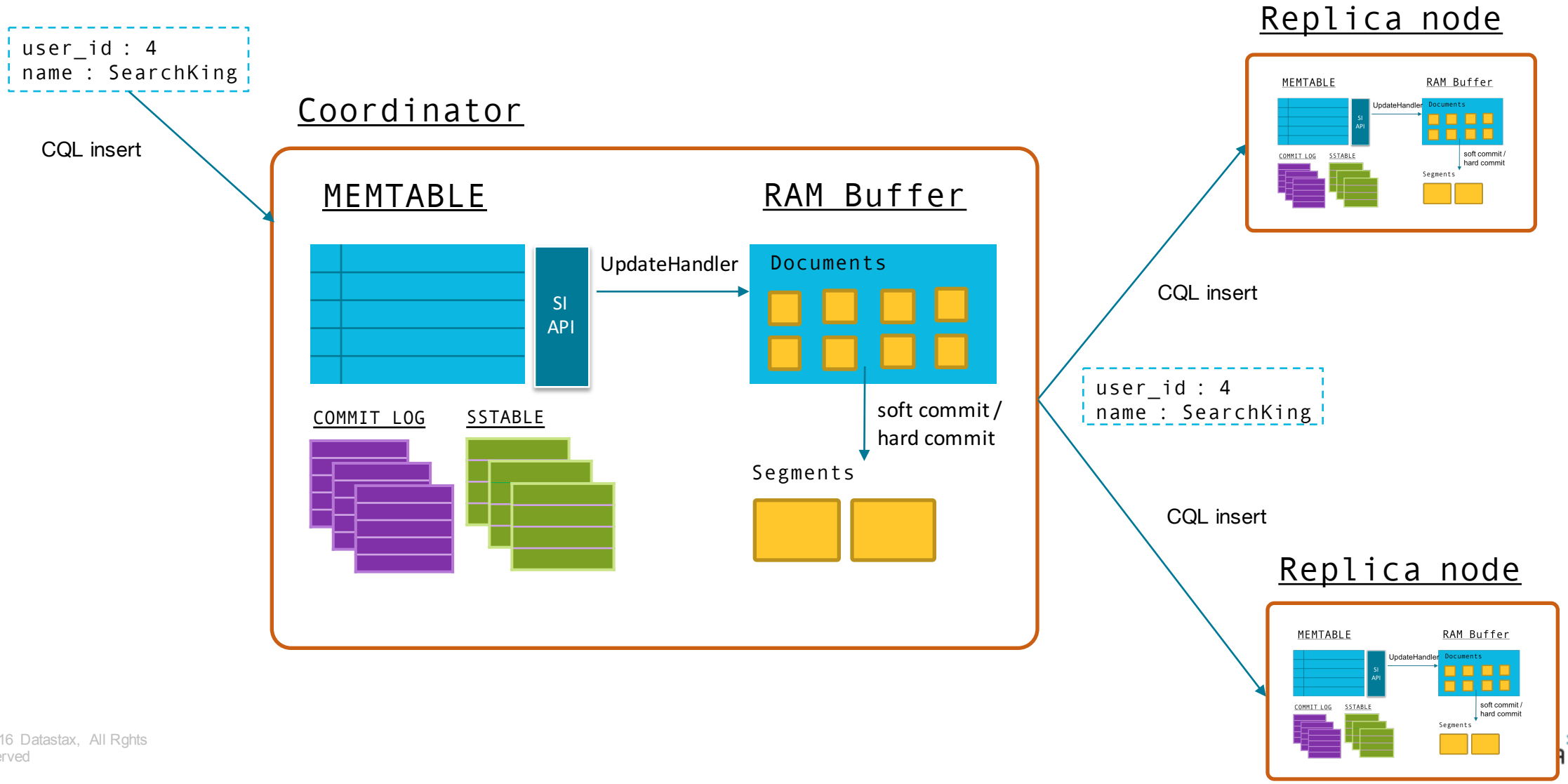
Complex application

Two seperated APIs, driver, read pathes

https://docs.datastax.com/en/datastax_enterprise/5.0/datastax_enterprise/srch/searchOssSolrDiff.html

DATASTAX

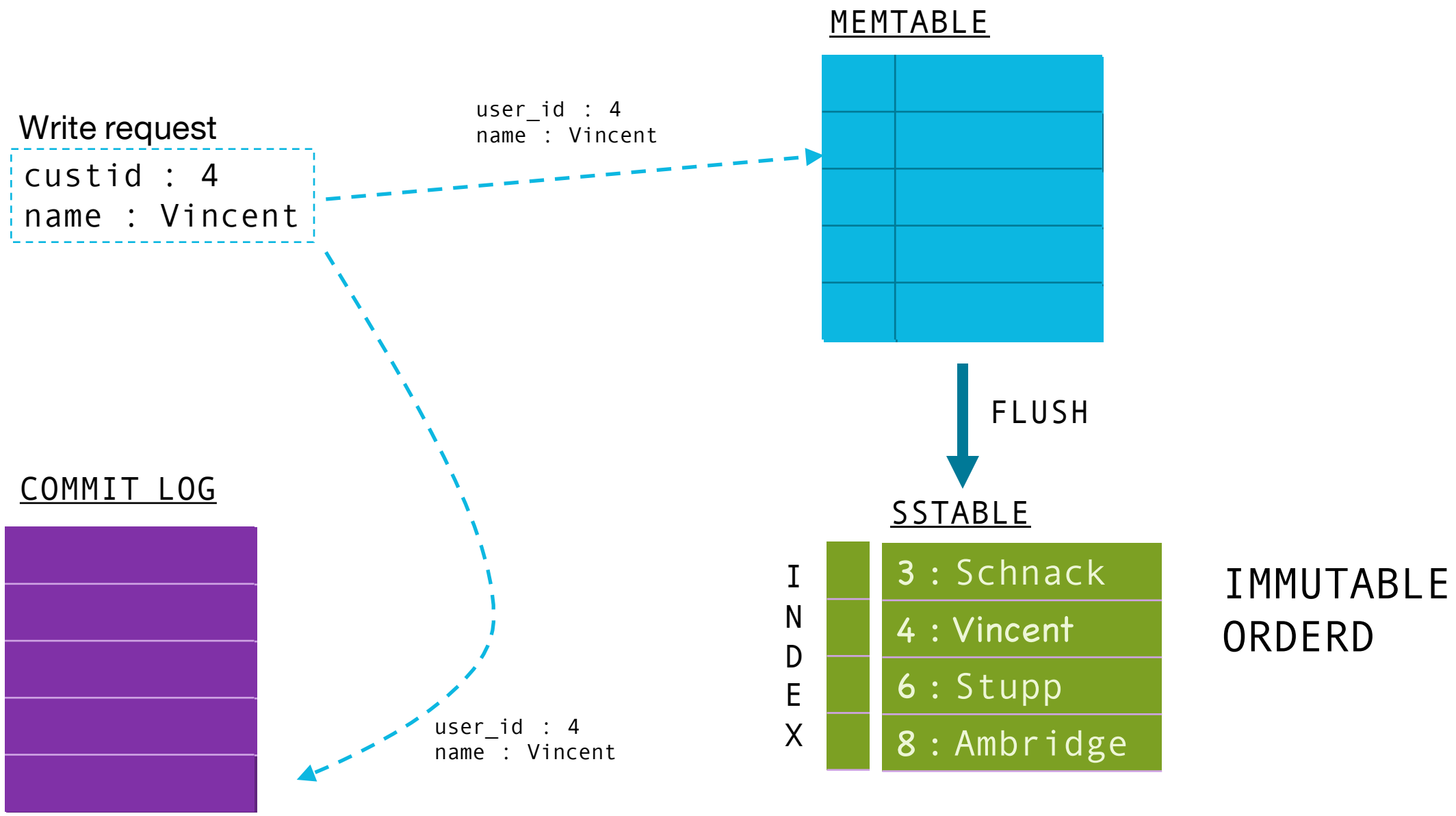
Inserting with DSE Search enabled

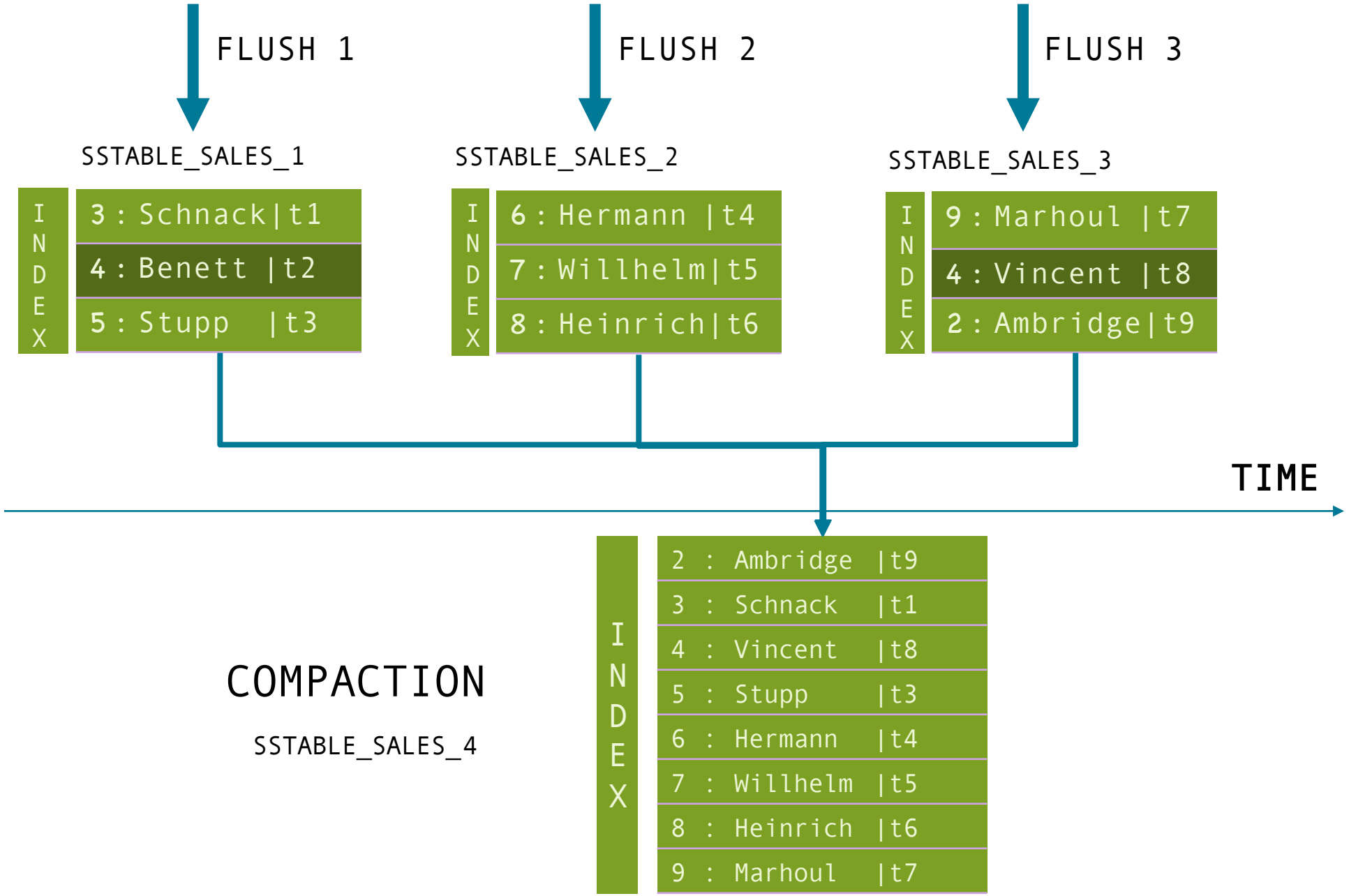


Smart Data Movement

Data Write Path, Tiered Storage and DSEFS

DATASTAX

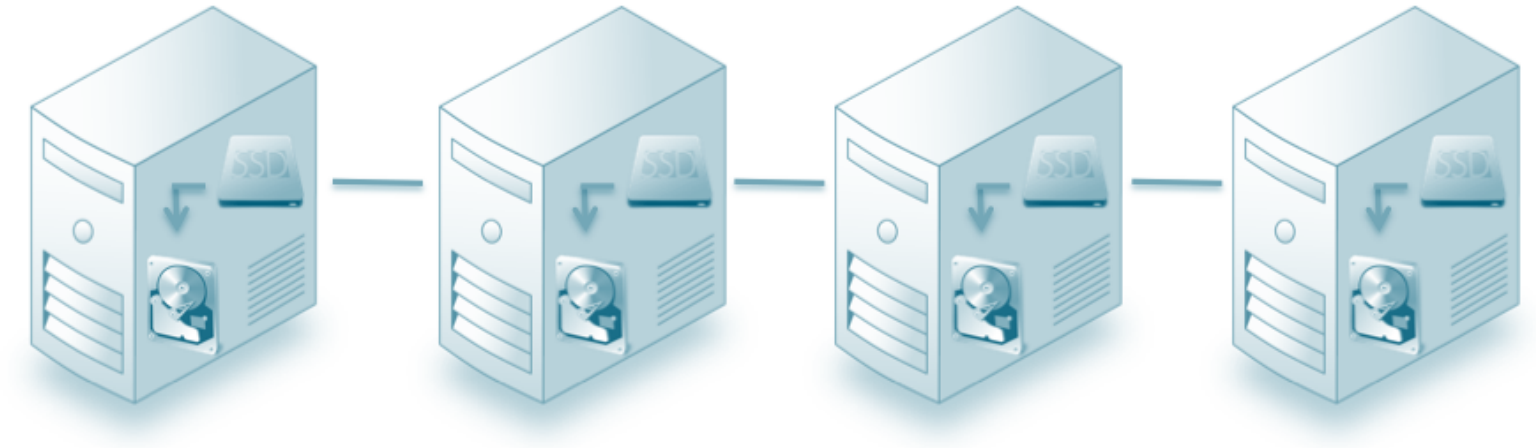

MEMTABLE
Write request
custid : 4
name : Vincent
user_id : 4
name : Vincent
FLUSH
COMMIT LOG
SSTABLE
INDEX
3 : Schnack
4 : Vincent
6 : Stupp
8 : Ambridge
IMMUTABLE
ORDERD
user_id : 4
name : Vincent


FLUSH 1
FLUSH 2
FLUSH 3
SSTABLE_SALES_1
SSTABLE_SALES_2
SSTABLE_SALES_3
INDEX
3 : Schnack|t1
4 : Benett |t2
5 : Stupp |t3
INDEX
6 : Hermann |t4
7 : Willhelm|t5
8 : Heinrich|t6
INDEX
9 : Marhoul |t7
4 : Vincent |t8
2 : Ambridge|t9
TIME
COMPACTION
SSTABLE_SALES_4
INDEX
2 : Ambridge |t9
3 : Schnack |t1
4 : Vincent |t8
5 : Stupp |t3
6 : Hermann |t4
7 : Willhelm |t5
8 : Heinrich |t6
9 : Marhoul |t7

DATASTAX

DSE Tiered Storage

- Able to automatically move data to different storage media based on defined criteria.
- Helps reduce storage costs by relegating lesser-used or older data to less expensive storage devices.
- Works on a granular per-row basis.



DATASTAX

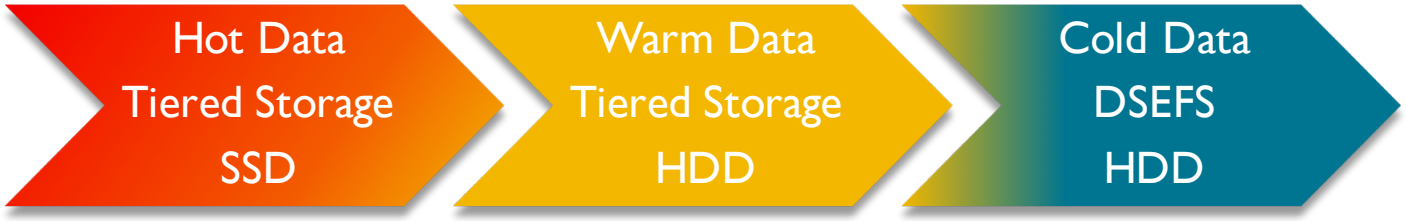
DSEFS

- Distributed file system, masterless, API compatible with HDFS
- Resiliency of metadata, being stored in Cassandra tables

- Cost effective, cold storage of data
- Staging
- Archiving
- Analytics (with Spark)

Storage Temperature Strategy

- Business value of data record per byte is low in IoT use cases
- Being able to optimize the cost of storage depending on the usage of the data is key
- A tiering / temperature management approach is a relevant response
  - **Hot Fast** storage using SSD for fresh data
  - Warm cost effective storage using HDD for older data (in-DB online archive)
  - Cold cheapest storage using file system for long term data (out-DB archive)
    - Can be used with Spark for analytical usages



DATASTAX

Use Cases

DATASTAX

DataStax Helps Macquarie Deliver Amazing Customer Experience

THE CHALLENGE: Drive digital transformation initiatives to enhance customer experience.

- Macquarie uses **DSE** to drive their **customer experience** initiative.
- Consolidated data from many disparate systems delivers **360º, real-time** customer visibility.
- Their world-class consumer banking app utilizes **real-time analytics** and **full text search.**
- Customers now have **better insight** into how they spent their money and where.
- Transformed from no retail presence to a digital consumer banking juggernaut in less than 2 years.

MACQUARIE

DATASTAX

GE PROVIDES PREDICTIVE MAINTENANCE INNOVATION WITH DATASTAX

THE CHALLENGE:

Collect sensor data from millions of devices from around the world and manage trillions of transactions per day.

- GE offers the first Industrial Cloud Platform called **GE Predix.** Datastax is part of the data services layer within the platform.
- DSE will collect sensor data from millions of devices from around the world to help GE provide **predictive maintenance** to their customers and increase **operational efficiencies**.
- Predix manages **trillions of transactions** per day. DSE was recognized as the only solution that could support this scale and data center replication.

WALMART DELIVERS THE NEXT GENERATION OF CUSTOMER EXPERIENCE WITH DATASTAX

THE CHALLENGE: Transform its strategy to avoid downtime on Black Friday and prevent food spoilage in grocery section of stores.

- Walmart serves nearly 260 million customers each week via their eCommerce websites and apps and 11,504 stores in 28 countries. With revenue of $486 billion in 2015, Walmart is transforming their strategy to deliver the next generation of customer experience.

- Product Catalog - DSE stores 100s of millions of Walmart products in their ecommerce site, and has helped them **avoid downtime for four Black Fridays in a row** since migrating from Oracle.
- IOT - DSE tracks chiller data to **prevent food spoilage**.

Walmart

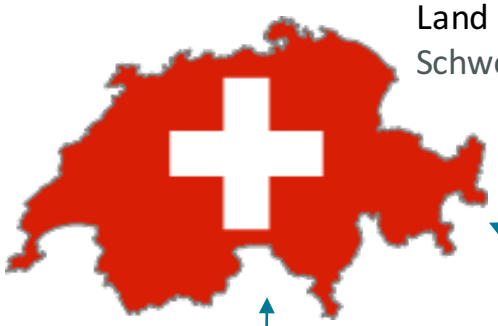
DATASTAX

Value in Relationship

Travers your Data

DATASTAX

Why Graph? “who invented it?”



Land
Schweiz

Geboren in
1704

produziert

Person

Werbung

geboren in
1962

Leonhard Euler
Mathematiker

“Wer hat‘s erfunden?“

gesprochen von

abgesetzt wegen

Schauspieler

Erich Vock

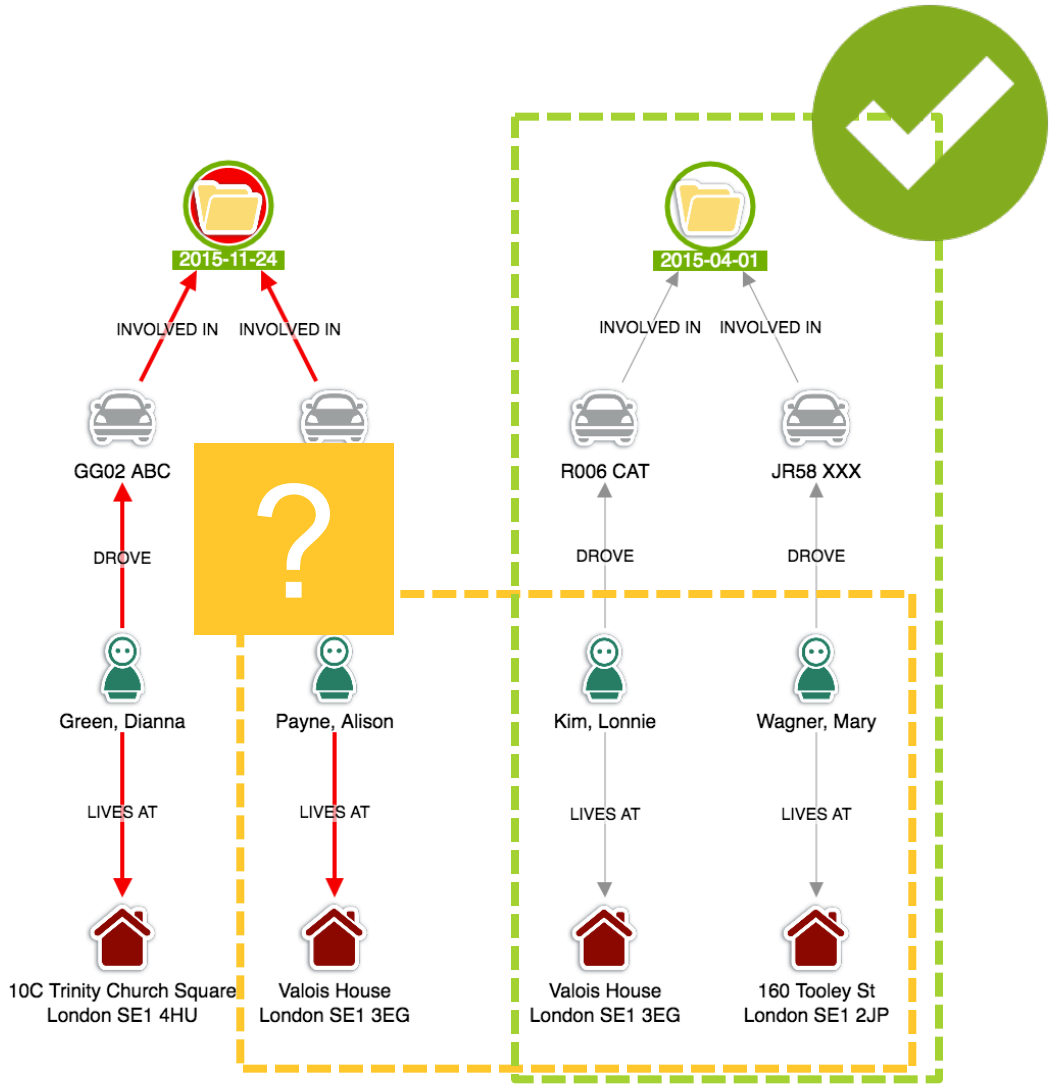
Markt
USA

DATASTAX

Graph to the rescue

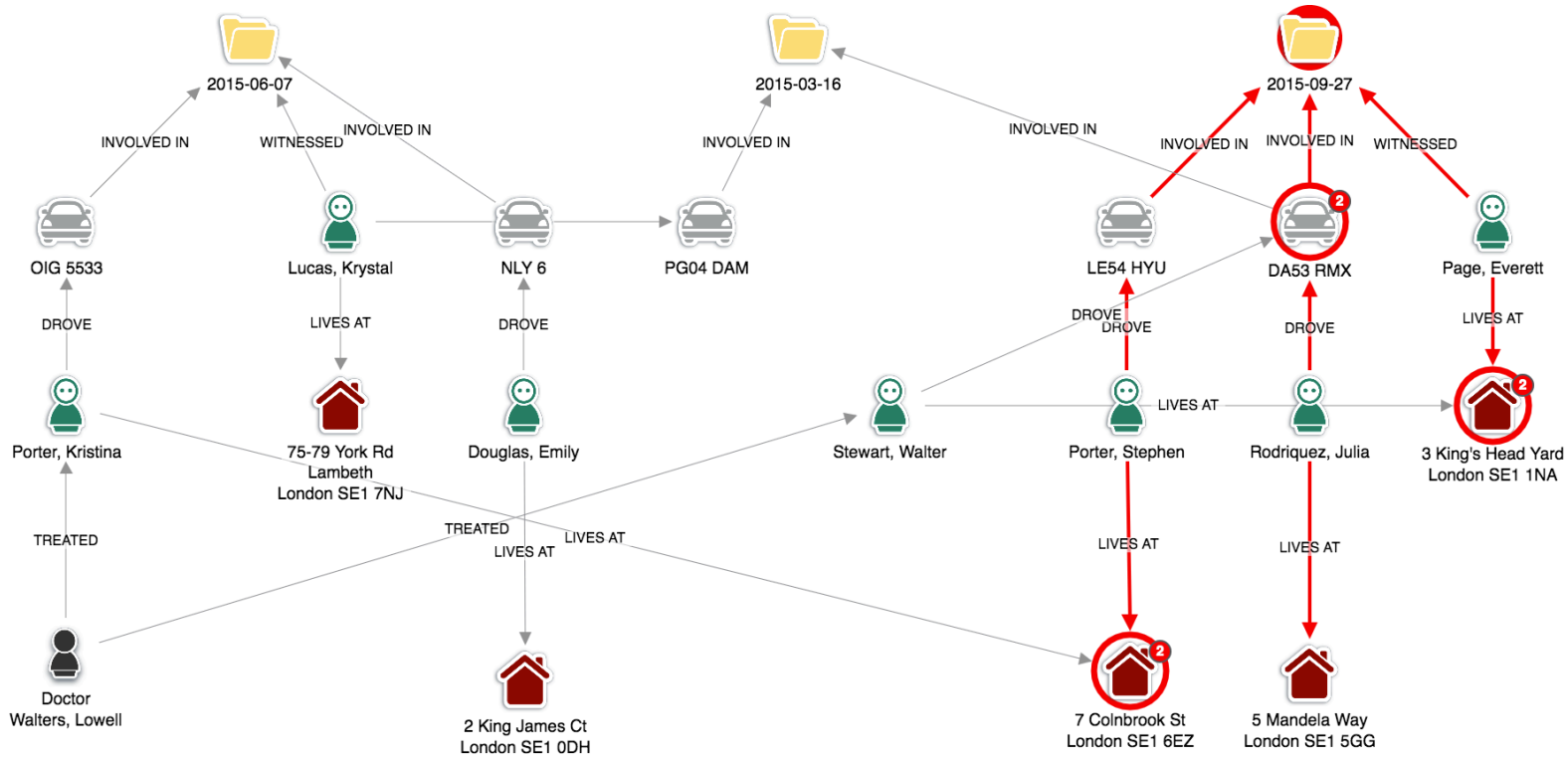
- In isolation, all of these transactions are ‘ok’
- So how can we identify fraud that doesn’t have obvious outliers?

- Defining a graph problem
  - Value is in the relationships
  - A large number of joins
  - Self-JOINs
- Questionable query performance



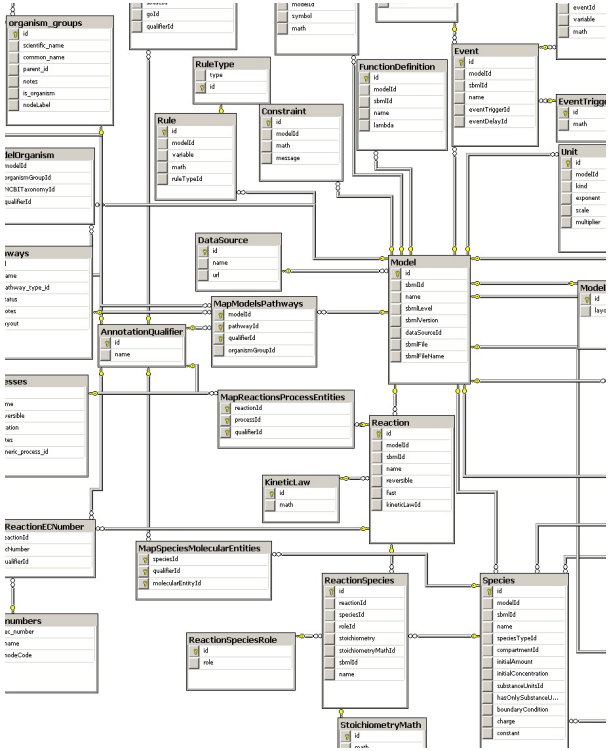
Fraud investigation

- Unusual connectivity -> “suspect” behavior
- Visualization helps humans uncover these anomalies
- Anomaly analysis & investigation

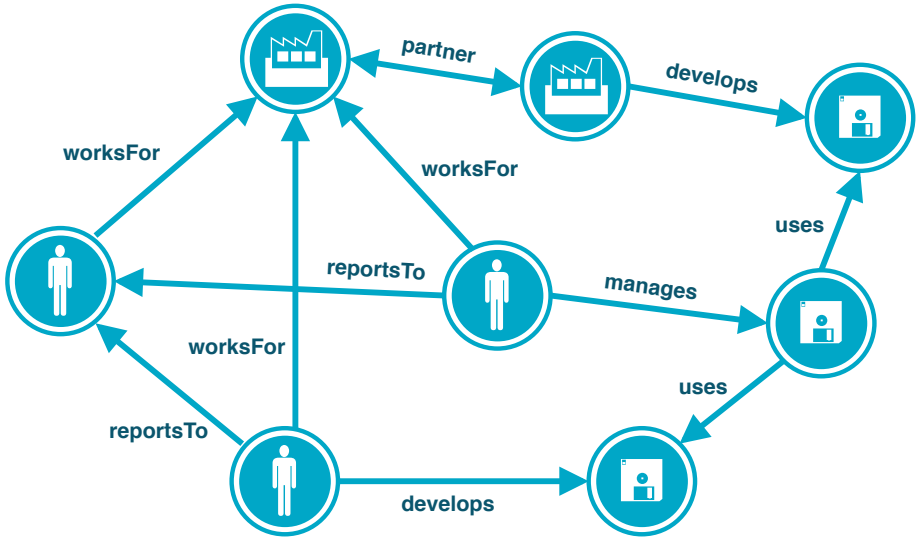


DATASTAX

RDBMS vs. Graph DB: Data Modeling

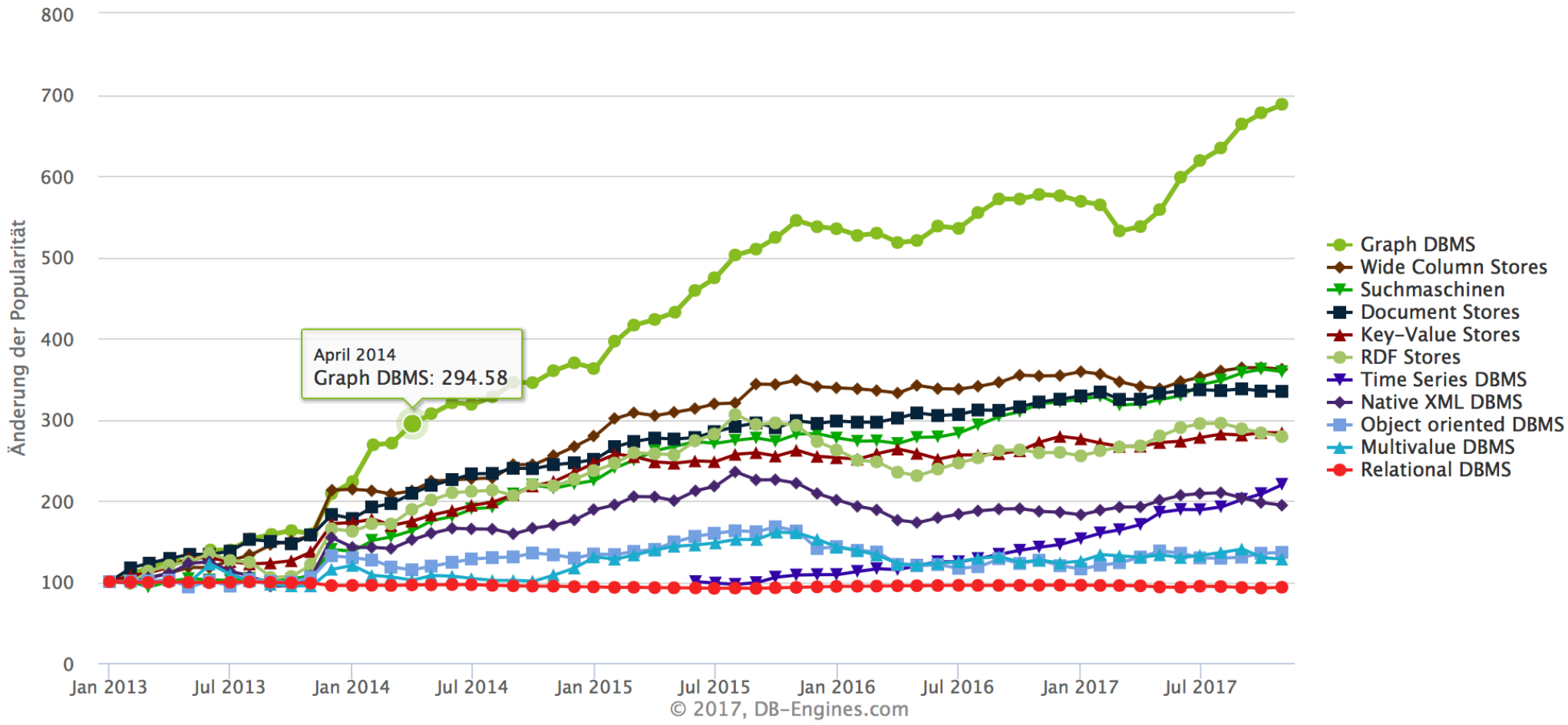


vs.



DATASTAX

Trends



DATASTAX

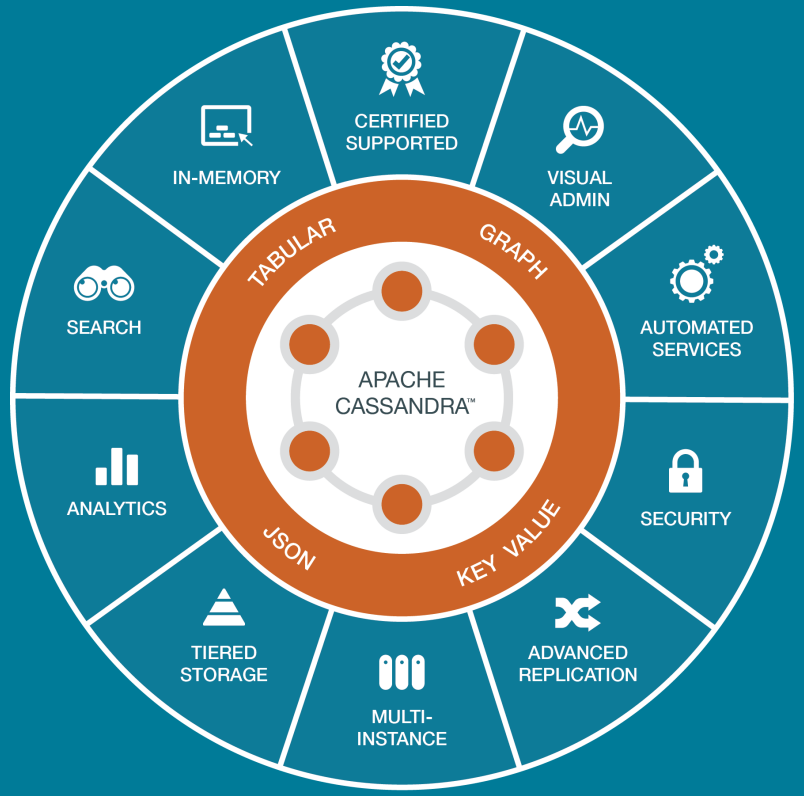
DSE Graph

A scalable, distributed graph database that is optimized for storing, traversing and querying complex graph data in real time

- Value data between relationships
- DSE Analytics and Search integrated
- Perfect for use cases:
  - Customer360,
  - Recommendations,
  - Fraud Detection

Questions?

DATASTAX



DATASTAX ENTERPRISE