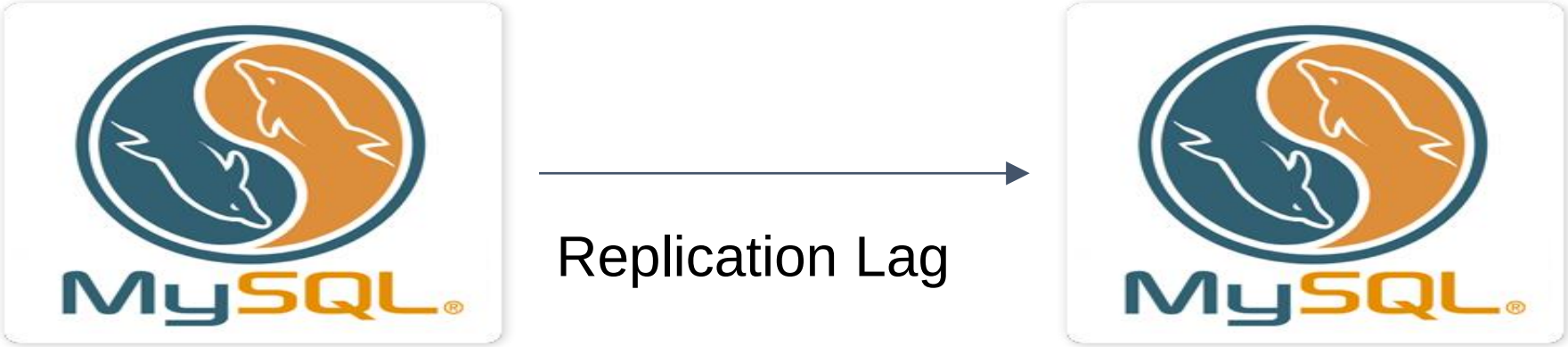
ADVANCED DATABASES CIS 6930
Dr. Markus Schneider

Group 2

Archana Nagarajan, Krishna Ramesh, Raghav Ravishankar, Satish Parasaram

Drawbacks of RDBMS



Master

Slave

- Vertical Scaling.
- ACID doesn’t always hold good for Big Data.
- Sharding becomes very difficult.
- High Availability is complicated.

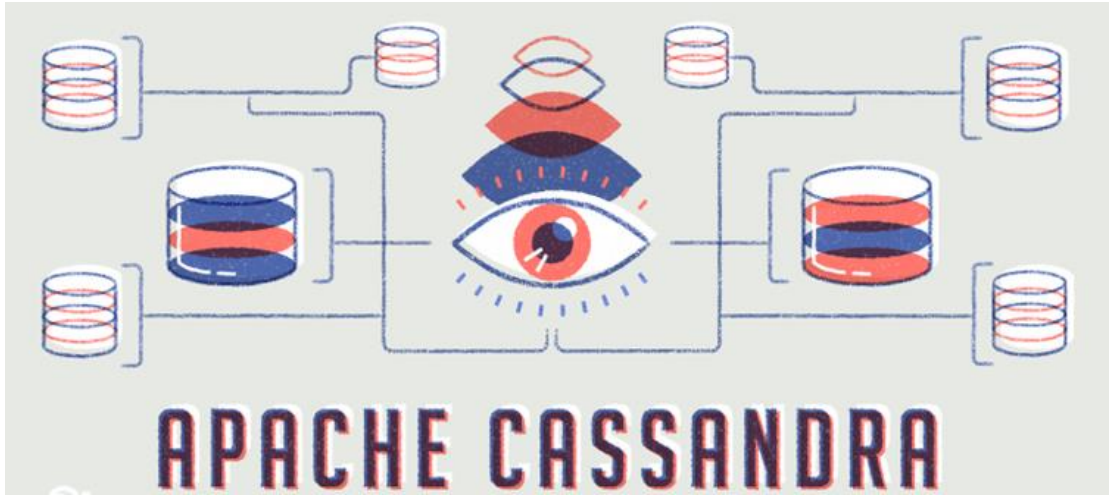
Why No SQL ?

- Non- Relational.
- Mostly Open Source.
- Easy Scalability and High Efficiency.
- No need to develop a detailed database model.
- Stores large volumes of data.
- Capable of performing agile operations.
- Object oriented programming.

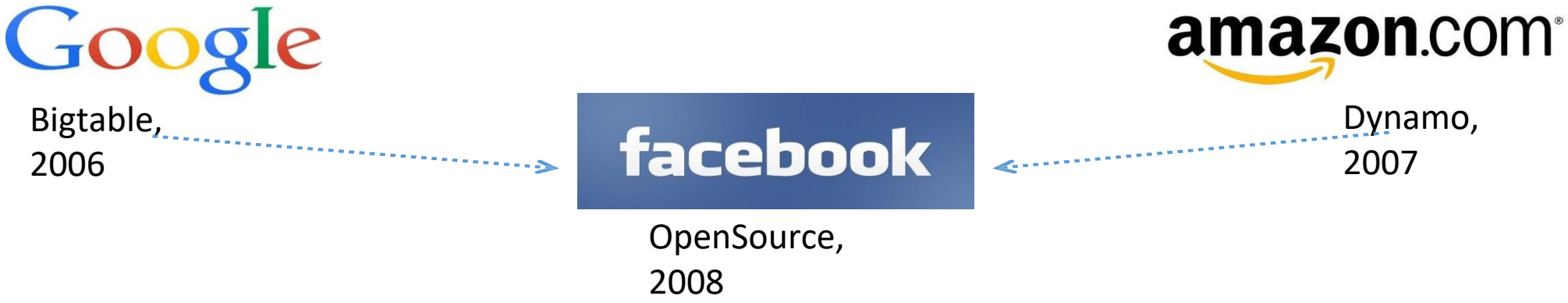
Definition of Cassandra

Cassandra is a

- Distributed
- High Performance
- Extremely scalable
- Fault tolerant
- Post relational database solution



History

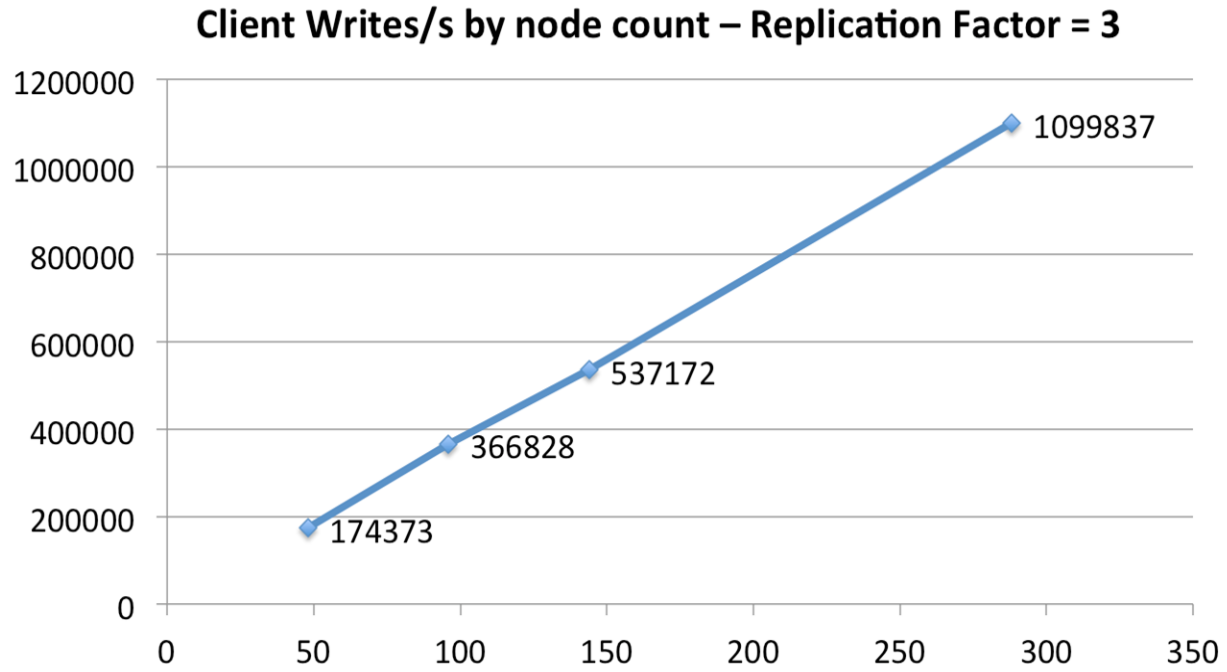


- Facebook released Cassandra as an open-source project on Google Code in July 2008.
- In March 2009 it became an Apache Incubator project.
- On February 17, 2010 it graduated to a top-level project.

How Big is Cassandra Today?

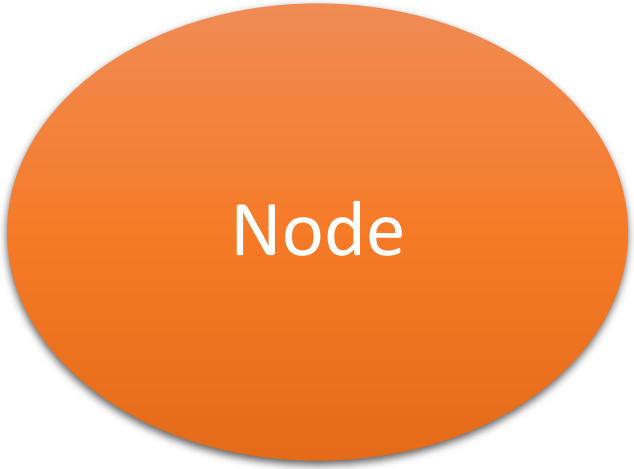
Key Features of Cassandra

- Distributed and Decentralized.
- Linear Scalability.
- Fault Tolerance.
- Handles huge amounts of data.
- Horizontal Scaling.
- Replication.
- Extremely Fast.



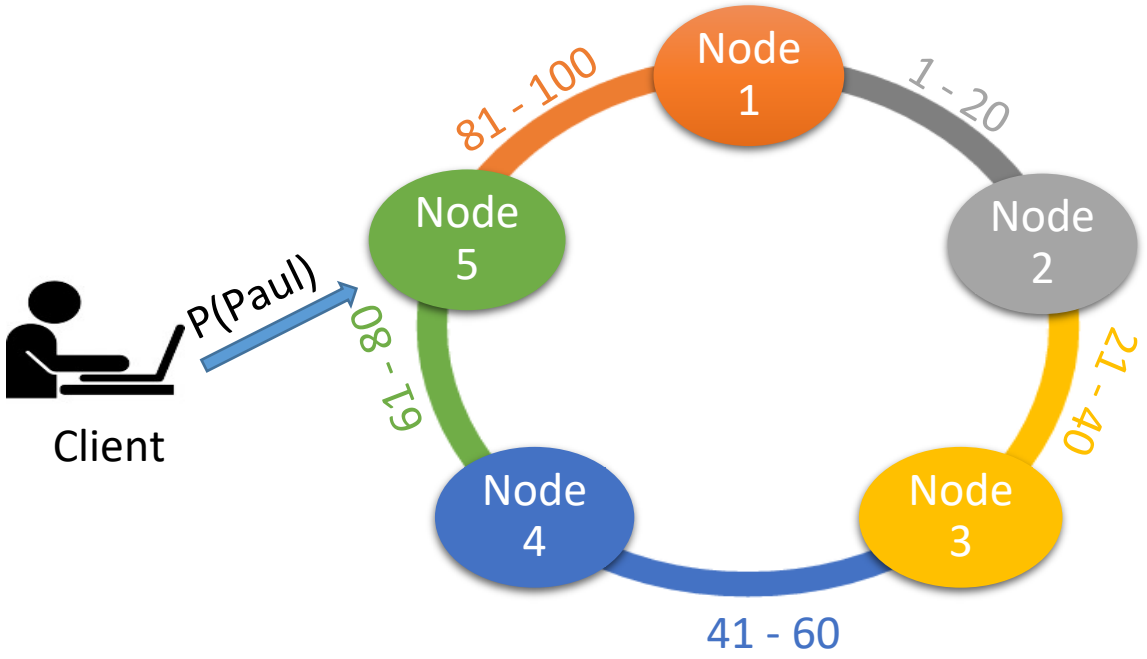
NETFLIX

Distributed Architecture: Node



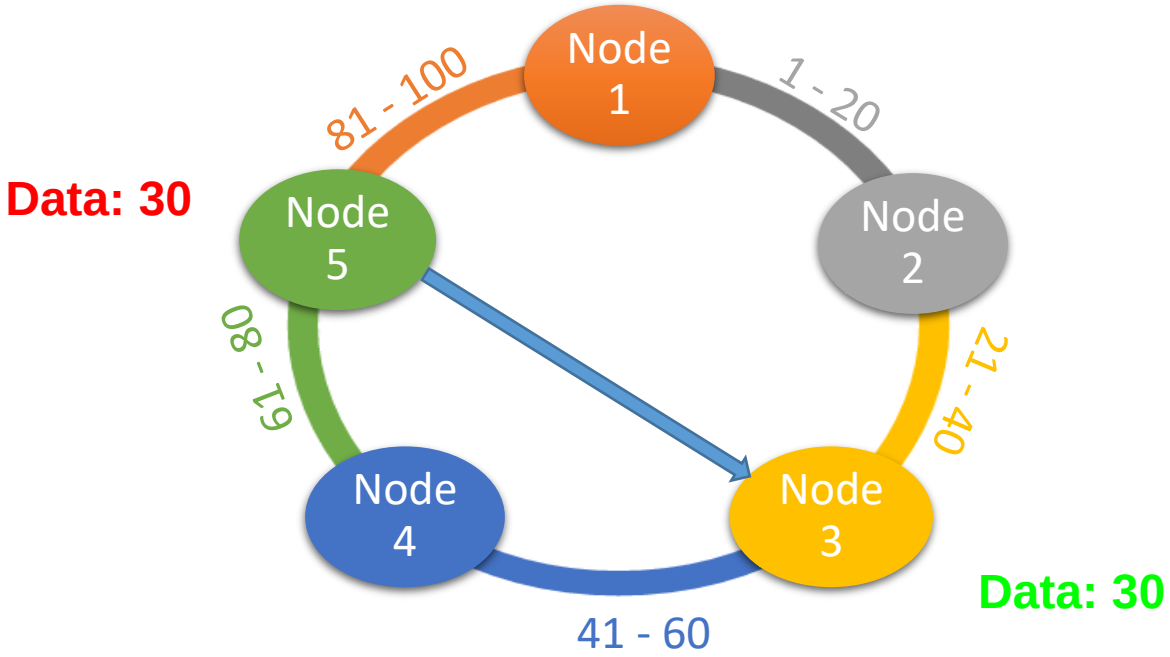
- Reads and Writes Data.
- Data Storage.
- Has an array of commands.

Distributed Architecture : Ring



- Clustering System.
- Token Ranges : -2^63 to 2^63-1.
- Partitioner.

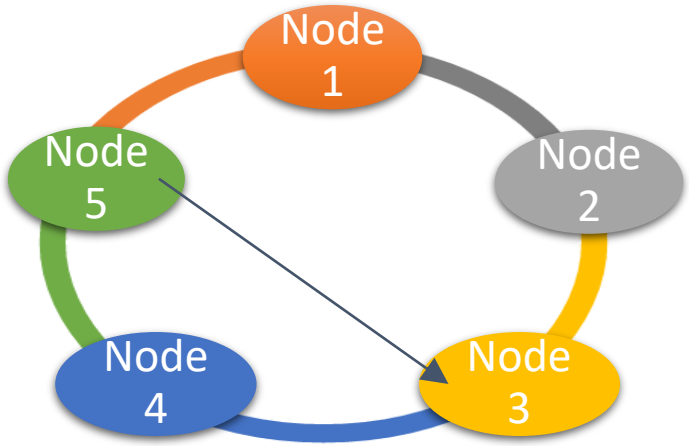
Distributed Architecture: Ring



- Co-ordinator.
- Four States : Joining , Leaving , Up and Down.
- Peer to Peer.
- V nodes.

Distributed Architecture:Gossip

Fig 1



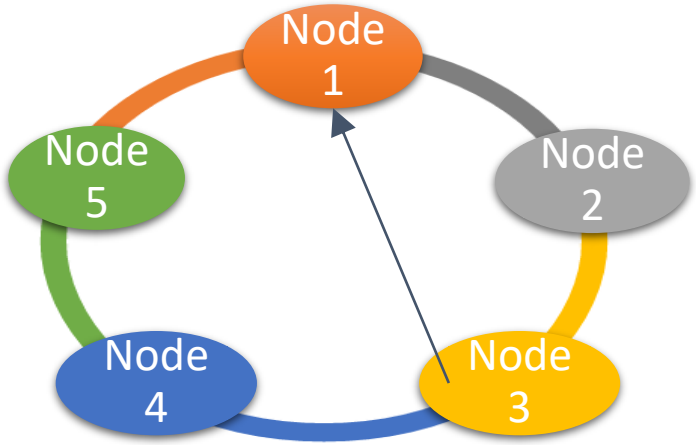


Fig 2

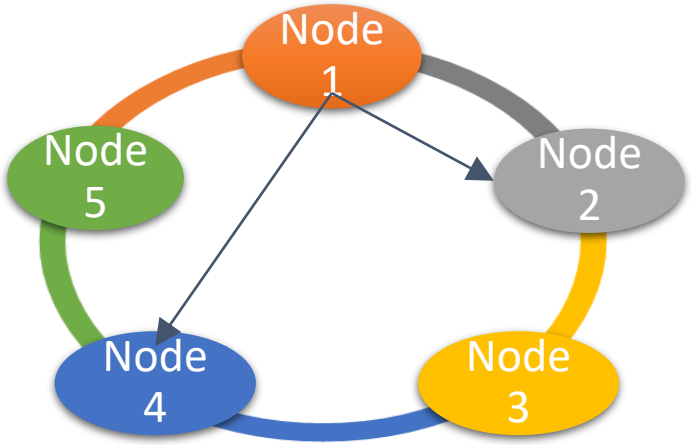
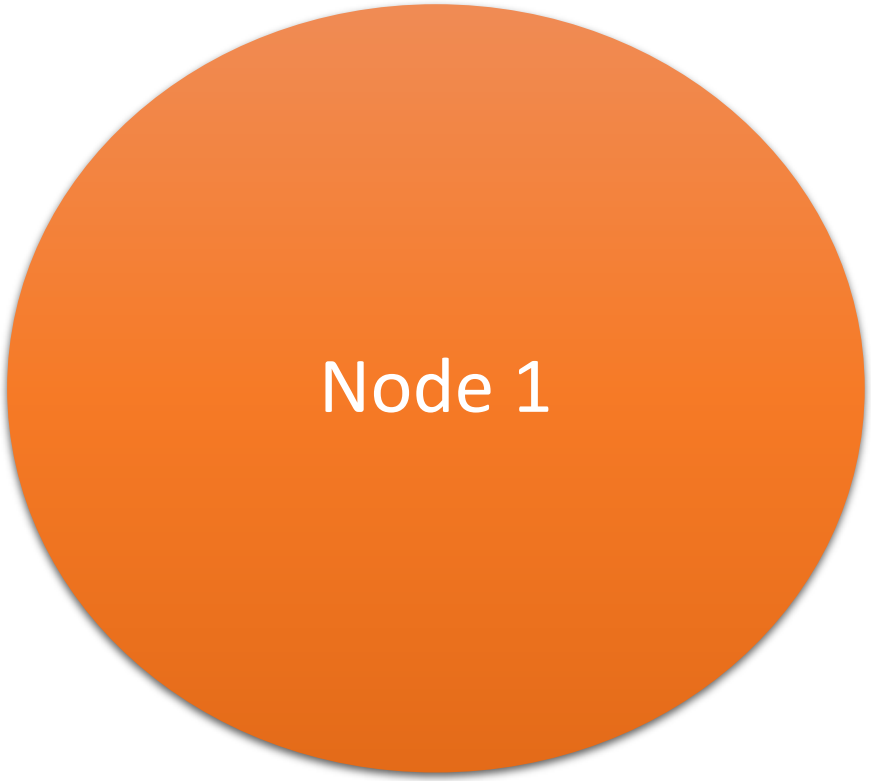


Fig 3

Information is thus, sent to all the nodes in the cluster.

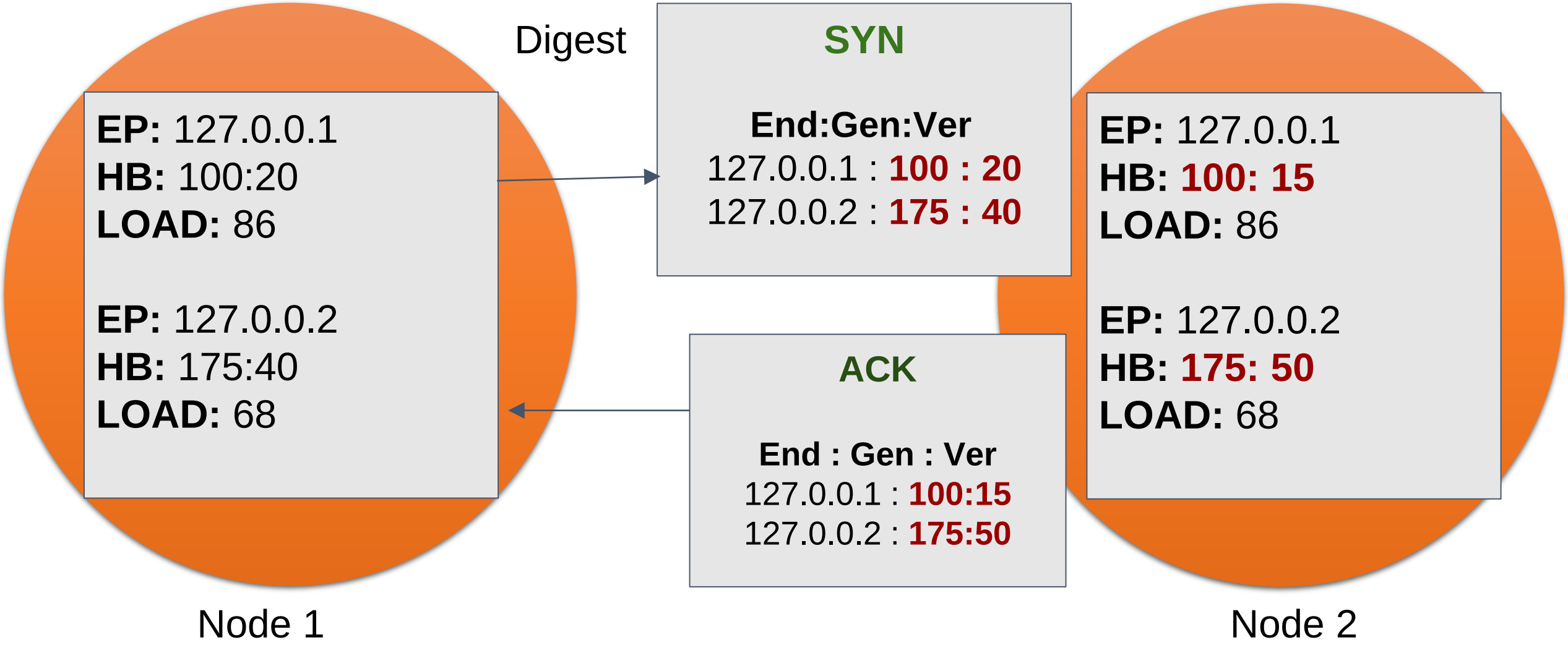
Distributed Architecture: Gossip



Endpoint state
Generation = 5
Version = 22

Application State:
Status = Normal
Dc = dc - west
Rack = rack1
Schema = c2a2b
Load = 100.0
Severity = 0.75

Distributed Architecture: Gossip



Distributed Architecture: Snitch

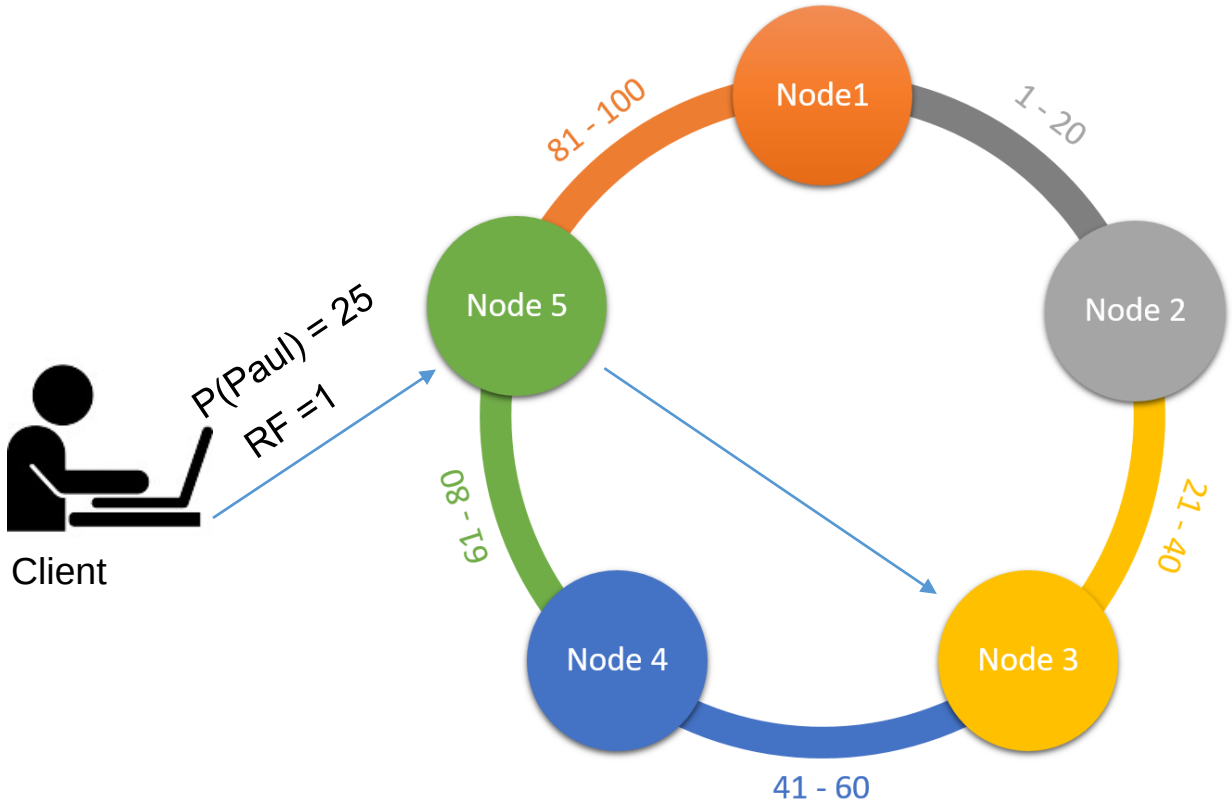
- Determines each node's rack and data center.
- The topology of the cluster.
- Configured in Cassandra.yaml.

There are mainly 2 groups of snitches. They are as follows:

1) Cloud Based Snitches.
2) Regular Snitches.

Characteristic Features of Cassandra

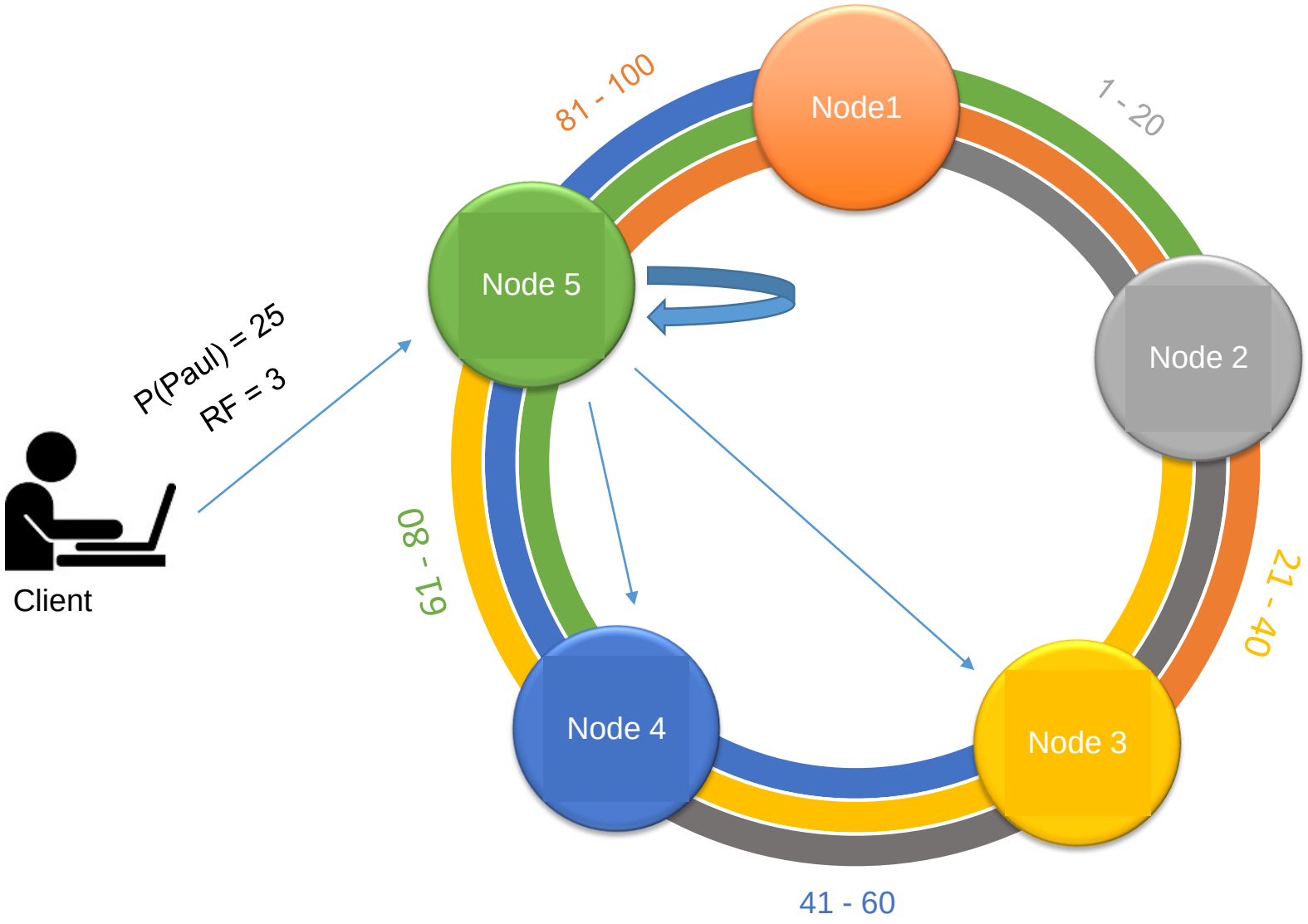
Replication Factor



- Similar to MySQL Sharding
- Point of Failure, Data loss
- Node reboot, network failure,
  Power loss, natural calamities
  Update patches
- What about RF=2?
- Simple Strategy

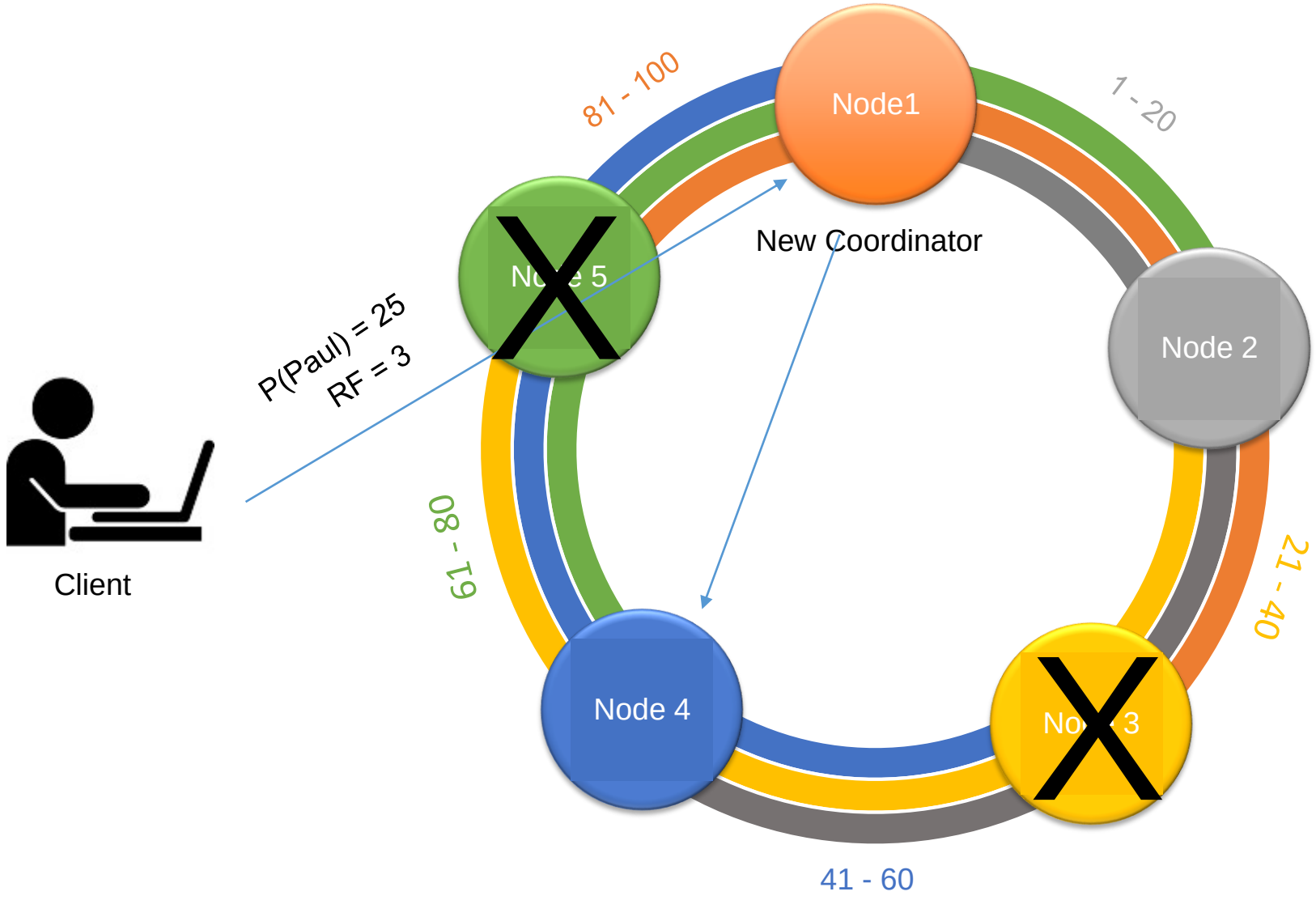
Create KEYSPACE socialdata
With REPLICATION = {
'class' : 'SimpleStrategy',
'replication_factor' : '1'
}

Replication Factor



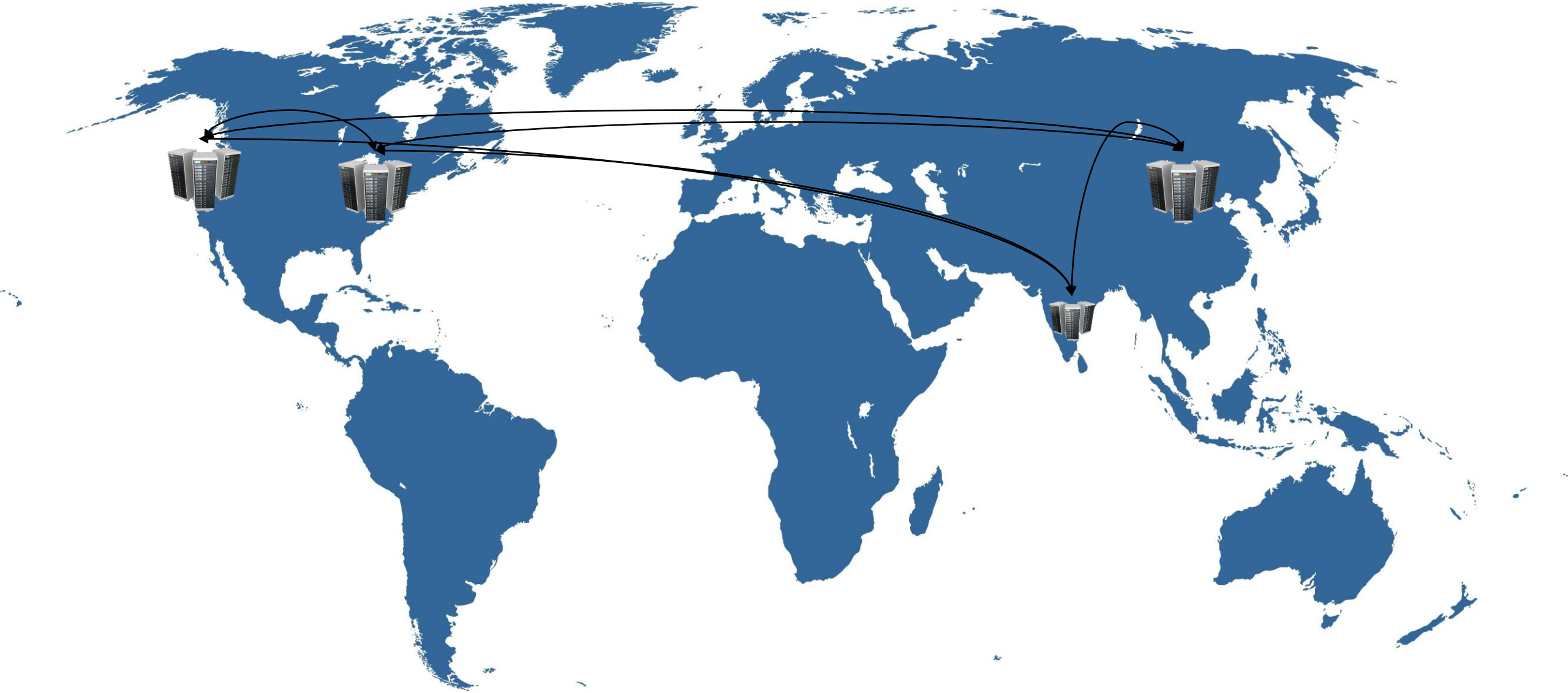
- Data replicated to 3 nearby nodes
- Node can be down
- Is RF=3 better than RF=2?
- Odd values of RF better
- RF = 3 used in production

Replication Factor



- Node 3, Node 5 are down
- Coordinator is down
- All nodes equally likely to be coordinator
- New coordinator randomly selected
- Data fetched from Node 4

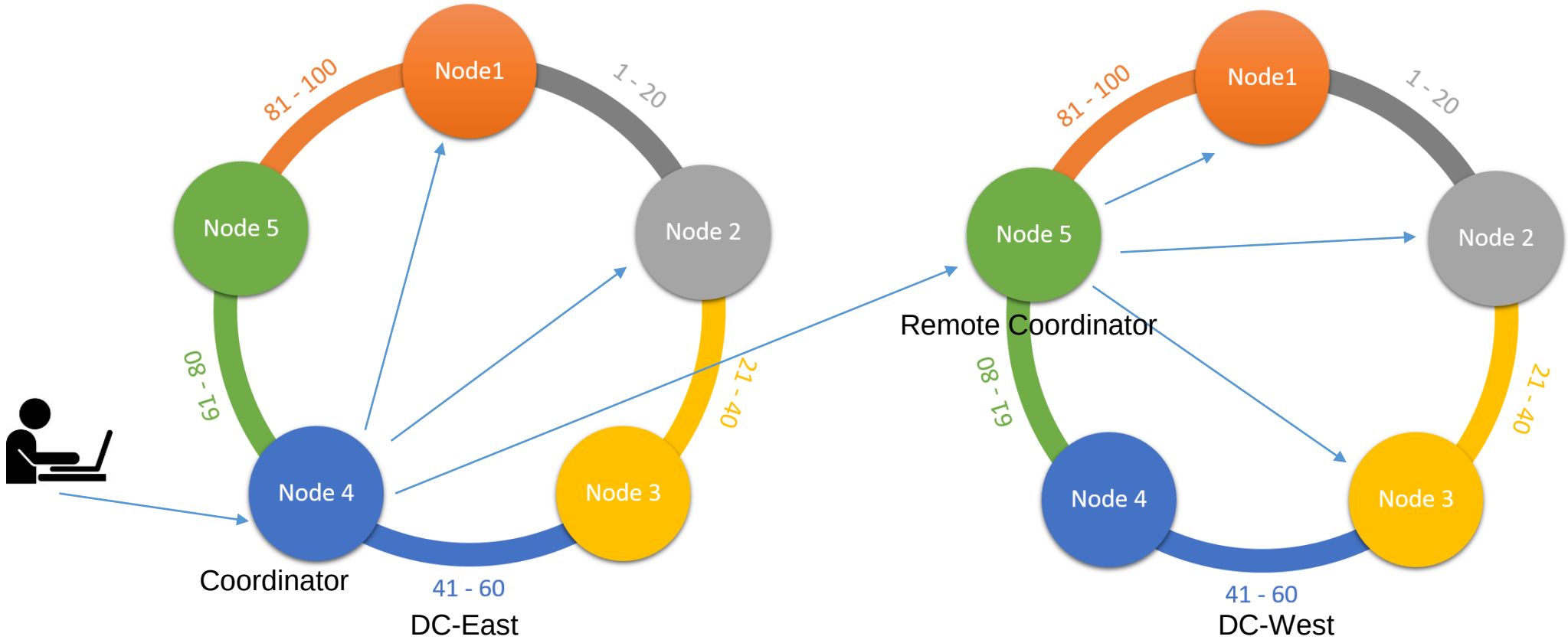
Network Topology Strategy



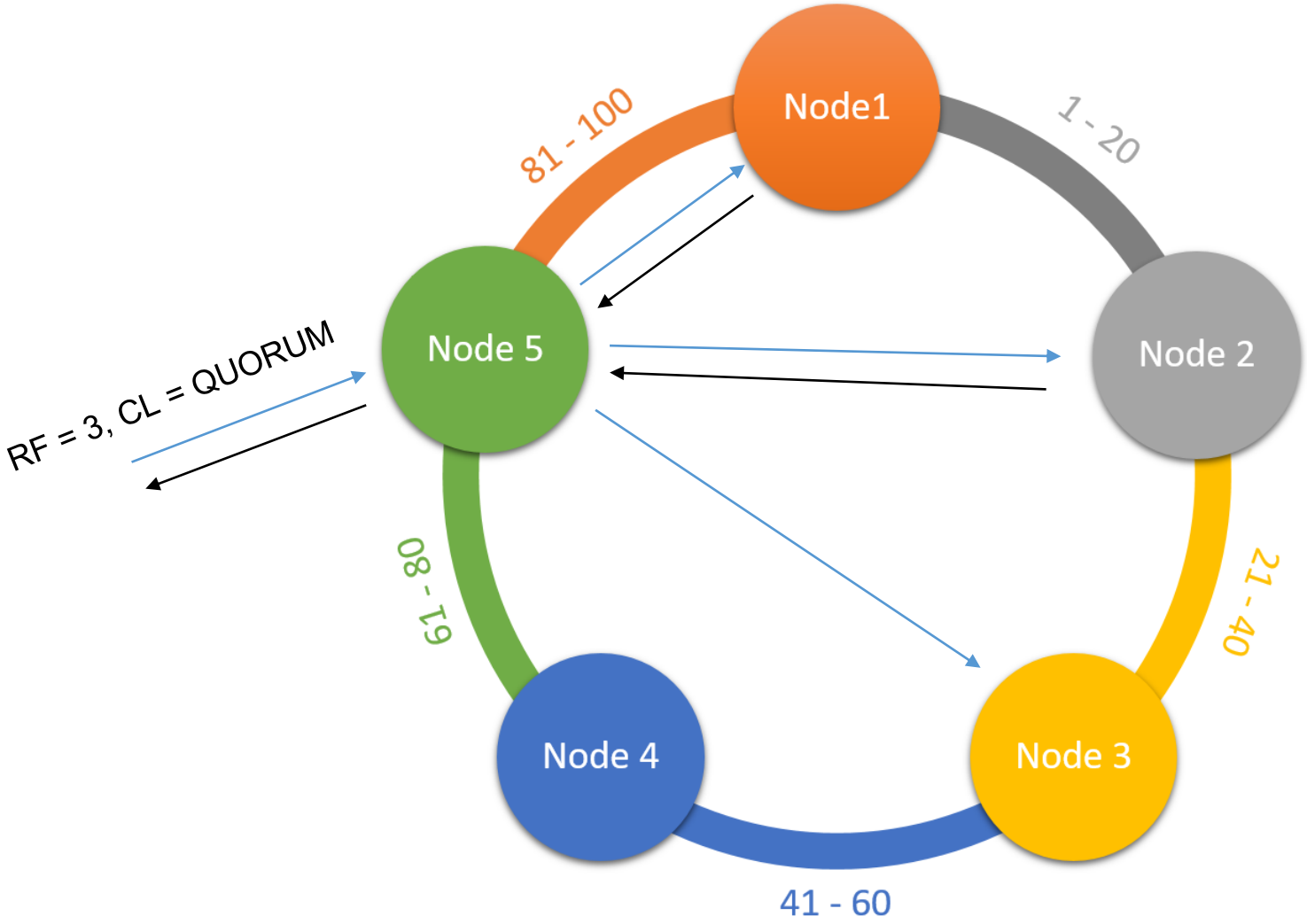
Replication Factor

Create KEYSPACE socialdata
With REPLICATION = {
'class' = 'NetworkTopologyStrategy', 'DC-East' : '2',
'DC-West' : '3' }

- Can lose a node
- Can lose an entire DC and be online
- Remote Coordinator
- High Availability
- cassandra.yaml



Consistency Levels



CL = 1
Coordinator chooses the closest node to respond, acknowledge

CL = QUORUM,
51% of replicas to respond back
RF = 2, RF = 3
49% of nodes can be down

CL = ALL, Strong consistency
If any node down, no data

Digest, Checksum

Consistency Levels in Writes and Reads
CL = ALL WRITE,
CL = ONE READ

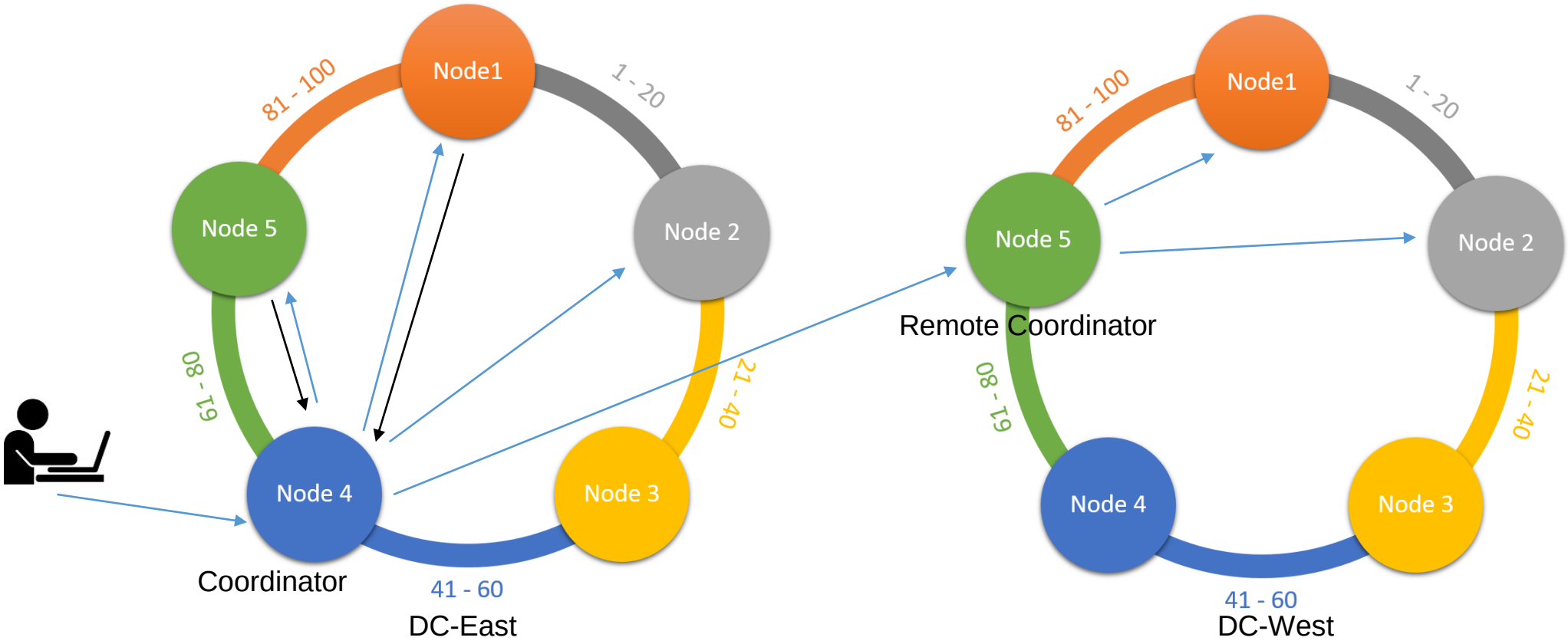
RF=3, CL = WRITE QUORUM, READ QUORUM, ATLEAST ONE OVERLAP

RF=3, CL=QUORUM ACROSS DCs leads to latency

Consistency Levels across data centers

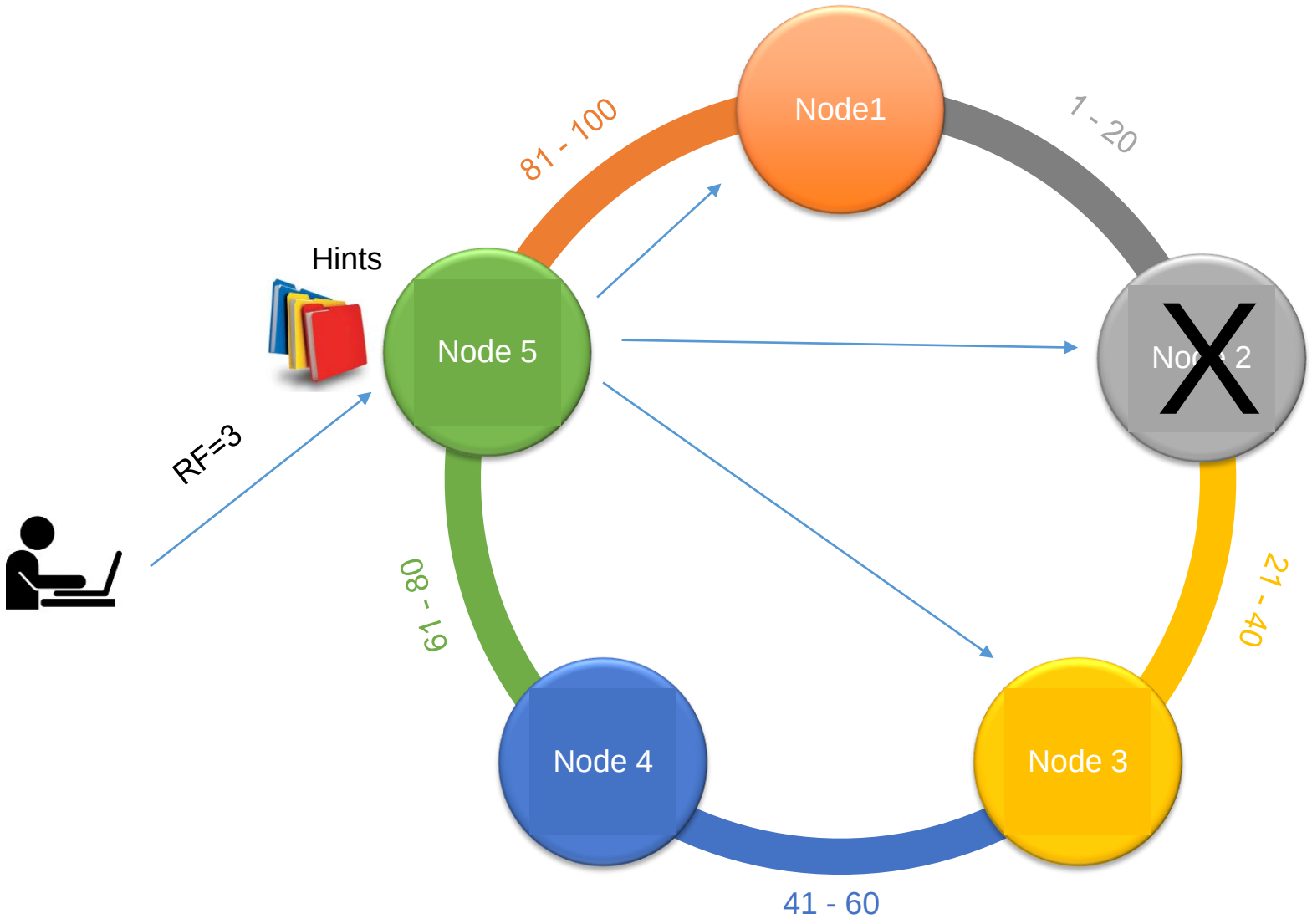
Create KEYSPACE socialdata
With REPLICATION = {
'class' = 'NetworkTopologyStrategy', 'DC-East' : '3',
'DC-West' : '2' }

CL = LOCAL_QUORUM,
QUORUM = latency of 100ms for response



Handoff

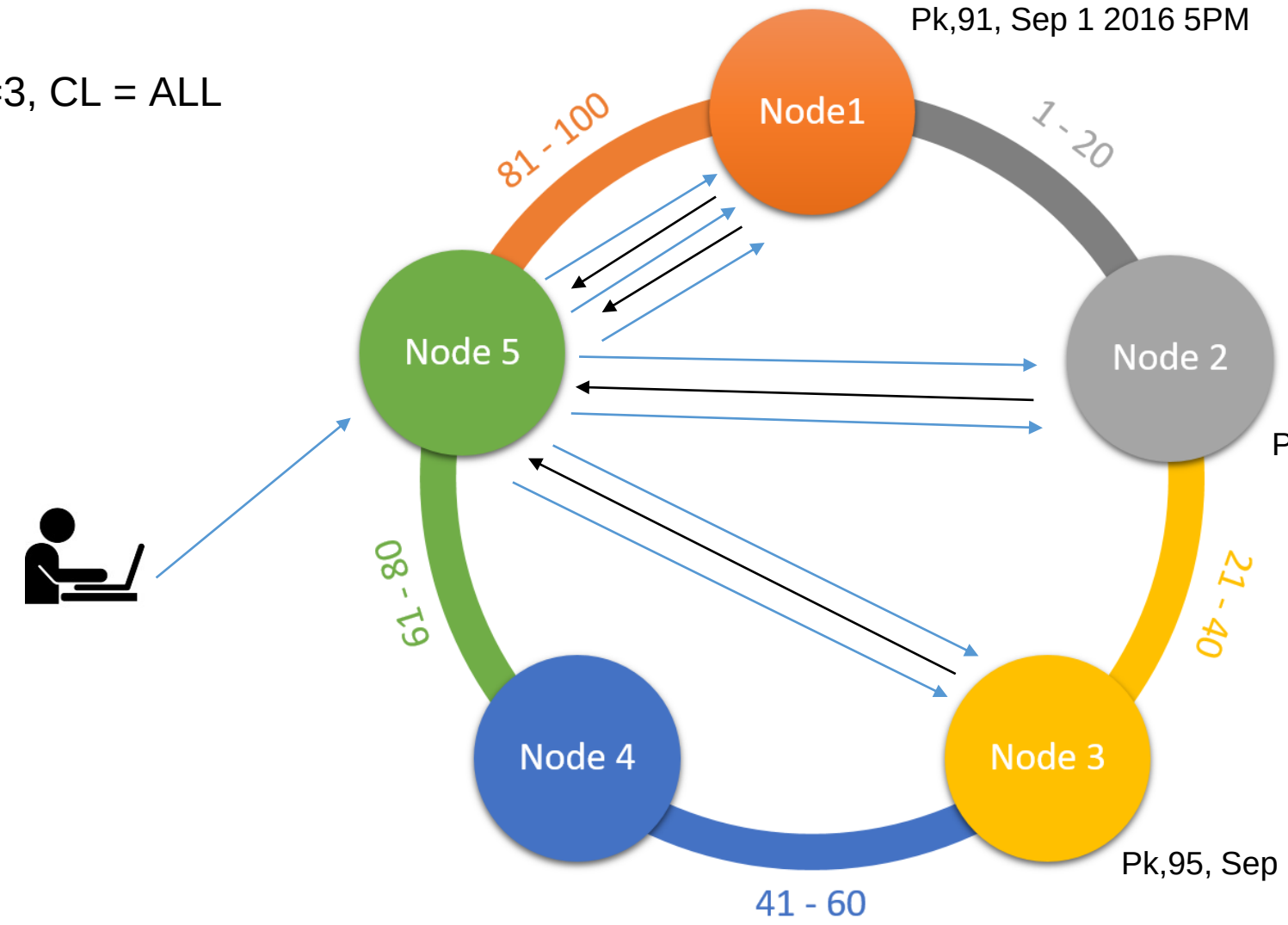
Hinted Handoff



- During Inserts, Updates, Deletes, RF=3
- Node 2 is down
- Inconsistency handled
- Hints are stored on the Coordinator, Node 5
- Node 2 is back up and Resyncs its data
- cassandra.yaml hinted_handoff default set to 3 hours

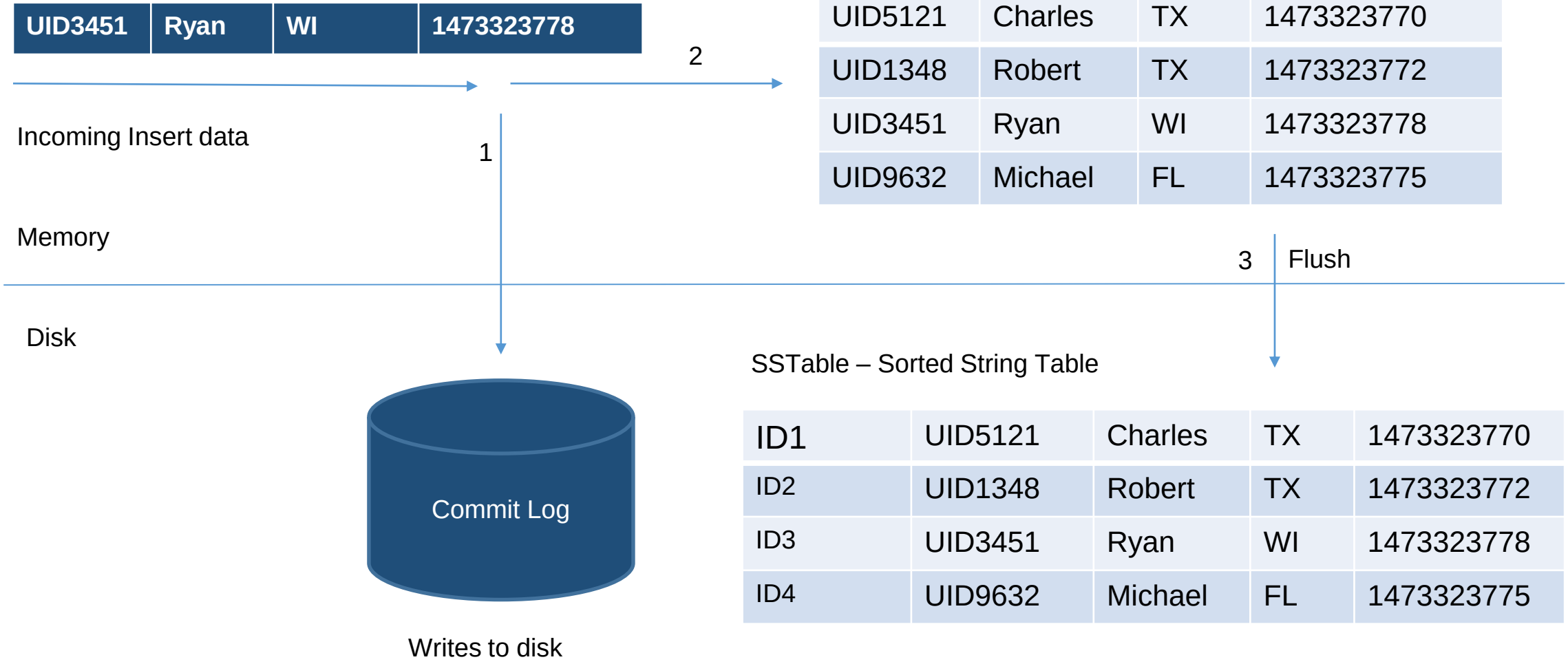
Read Repair

RF=3, CL = ALL



- Select operation
- Digest and checksum dint match
- Request for timestamp
- Update nodes with latest value based on timestamp

Cassandra Write Path



Memtable - Partition Key 58

UID5121	Charles	TX	1473323770
UID1348	Robert	TX	1473323772
UID3451	Ryan	WI	1473323778
UID9632	Michael	FL	1473323775

SSTable – Sorted String Table

ID1	UID5121	Charles	TX	1473323770
ID2	UID1348	Robert	TX	1473323772
ID3	UID3451	Ryan	WI	1473323778
ID4	UID9632	Michael	FL	1473323775

Cassandra Write Path

- Updates
- Deletes
- Tombstones
- gc_grace_seconds

Memtable – Partition Key 58

UID2101	Kevin	FL	1473323780
UID2109	Richard	FL	1473323782
UID3451	Ryan	NY	1473323788
UID2191	Steven	CA	1473323785

Memory

Flushed

Disk

SSTable 1

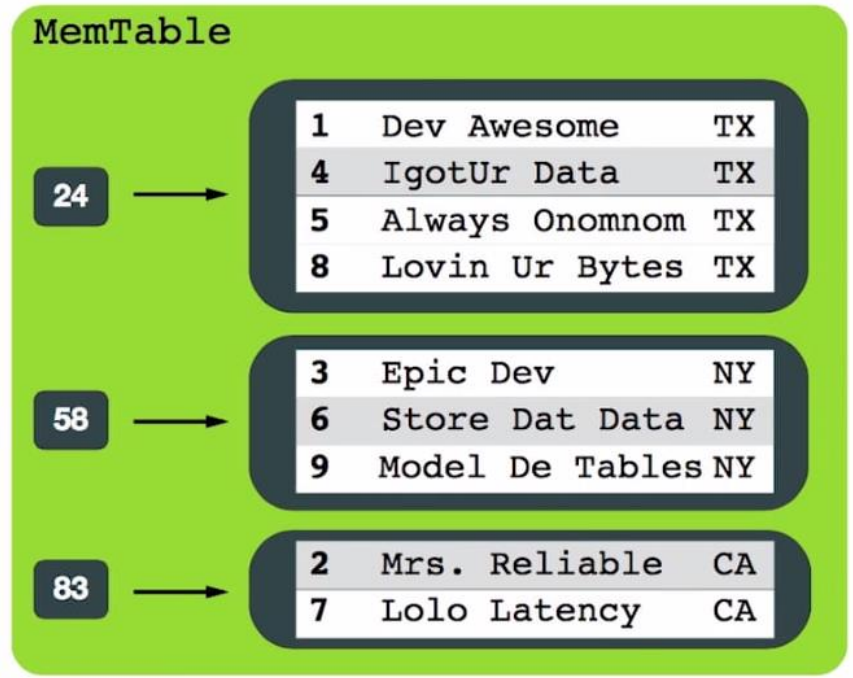
ID1	UID5121	Charles	TX	1473323770
ID2	UID1348	Robert	TX	1473323772
ID3	UID3451	Ryan	WI	1473323778
ID4	UID9632	Michael	FL	1473323775

SSTable 2

ID5	UID2101	Kevin	FL	1473323780
ID6	UID2109	Richard	FL	1473323782
ID7	UID3451	Ryan	NY	1473323788
ID8	UID2191	Steven	CA	1473323785

Cassandra Read Path

58



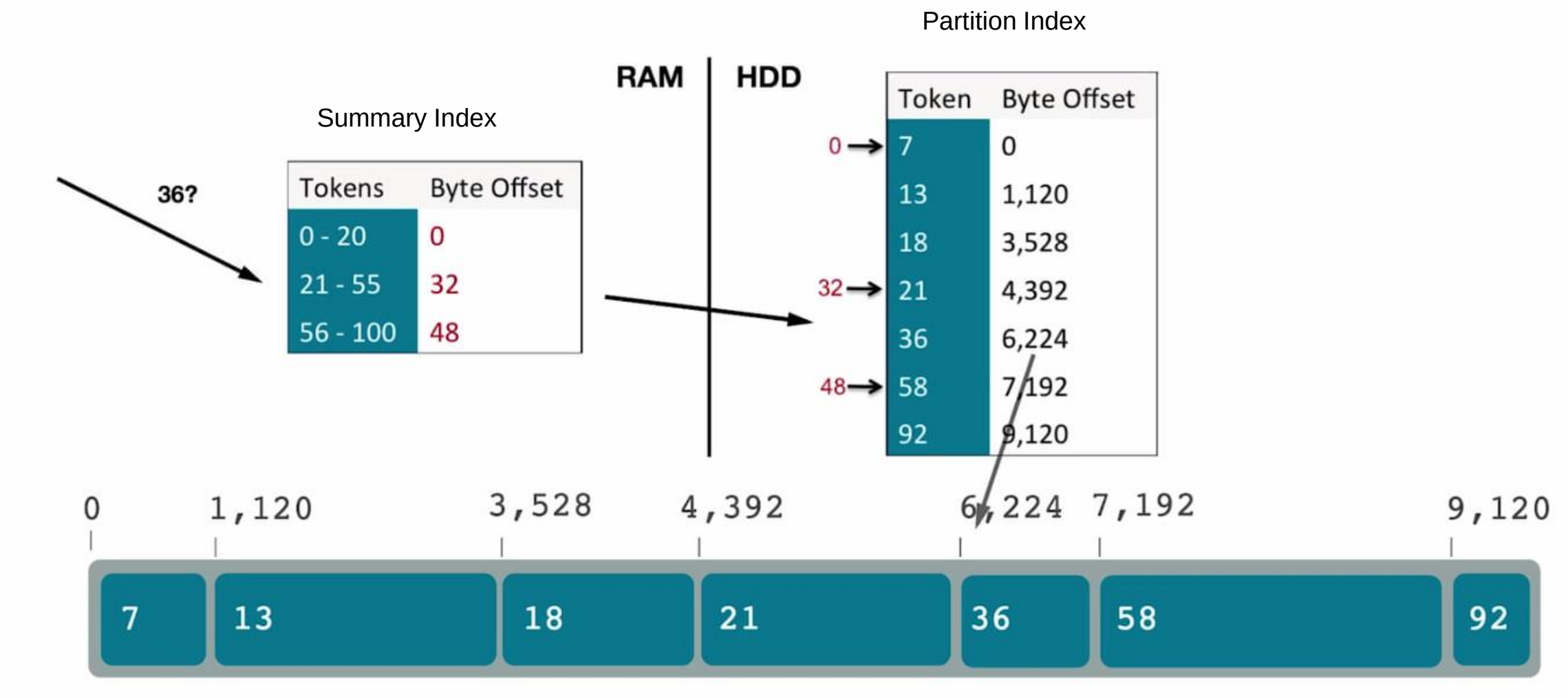
SSTable 1

SSTable 2

SSTable 3

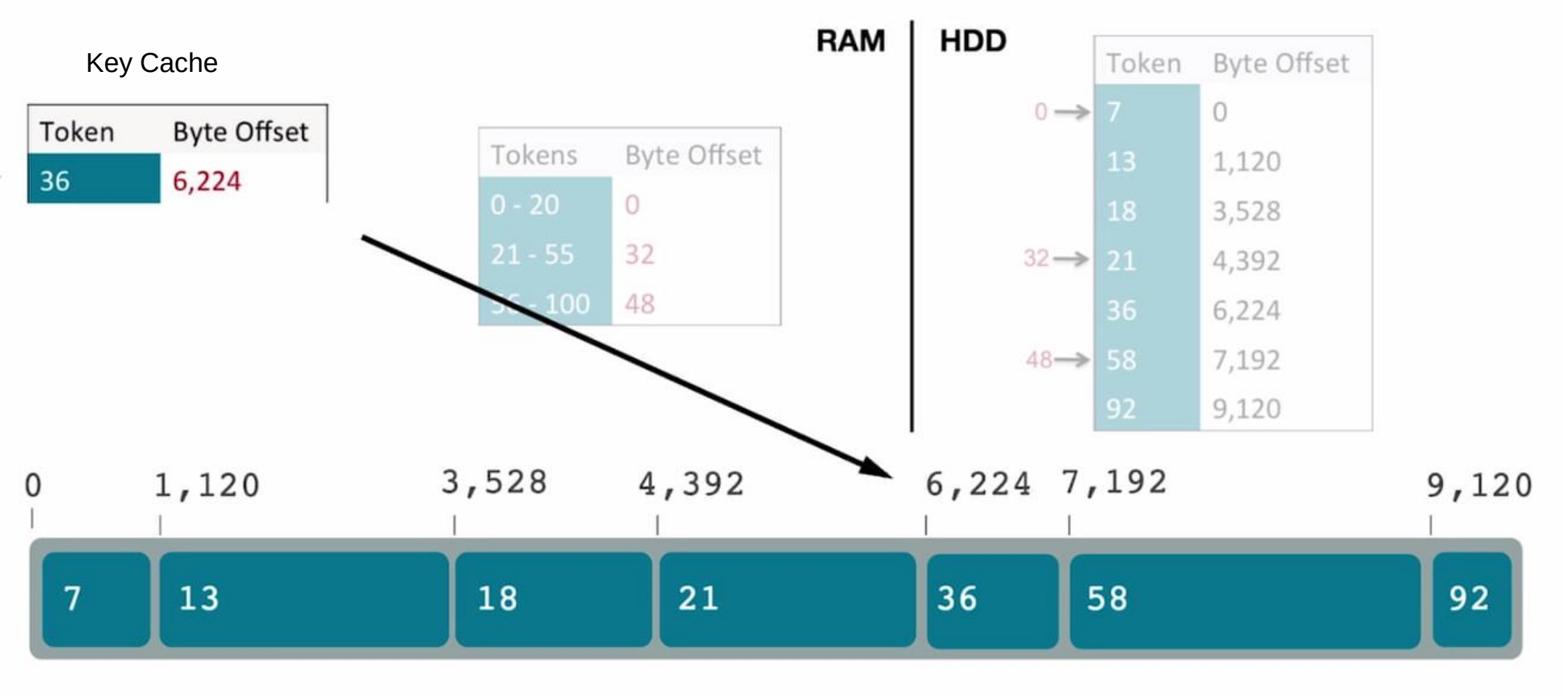
Reference : http://www.datastax.com

Cassandra Read Path - SSTable



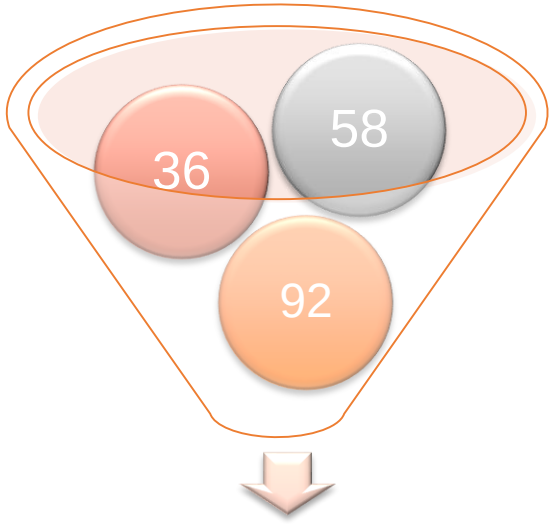
Reference : http://www.datastax.com

Cassandra Read Path - SSTable



Reference : http://www.datastax.com

Cassandra Read Path - SSTable



Bloom Filter

- Determines probabilistically if a value is not in a SSTable
- Gives false positives but zero false negatives
- Eliminates need to search across multiple SSTables

SSTable 1

SSTable 2

SSTable 3

Compaction

SSTable 1

1	Johnny (92)
2	Betsy (49)
3	Nicholi (85)
4	Sue (41)
5	Sam (96)

SSTable 2

1	Johnny (181)
2	(X) (176)
3	Norman (148)
5	(X) (184)
6	Henrie (134)

SSTable 3

Reference : http://www.datastax.com

Compaction

SSTable 1

1	Johnny (92)
2	Betsy (49)
3	Nicholi (85)
4	Sue (41)
5	Sam (96)

SSTable 2

1	Johnny (181)
2	(X) (176)
3	Norman (148)
5	(X) (184)
6	Henrie (134)

SSTable 3

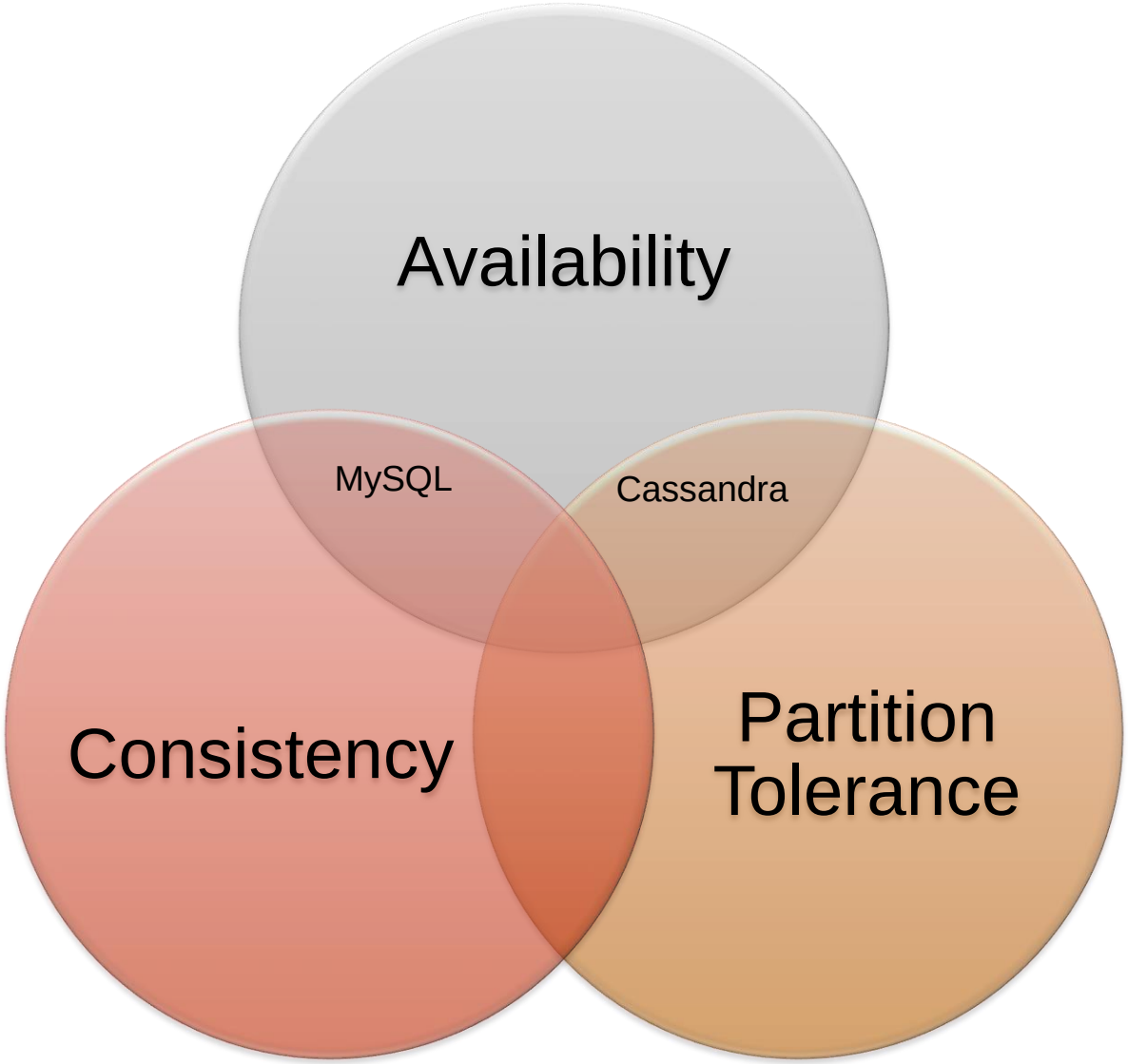
1	Johnny (181)
3	Norman (148)
4	Sue (41)
5	(X) (184)
6	Henrie (134)

Reference : http://www.datastax.com

Compaction Strategy

- min_sstable_size 50Mb
- min_threshold 4 – Minimum number of SSTables required for compaction
- max_threshold 32 – Maximum number of SSTables allowed for compaction
- tombstone_compaction_interval – 86400secs

CAP Theorem


Availability
MySQL
Cassandra
Consistency
Partition
Tolerance

Data Model

Column Oriented DBs

- The storage of data is column value wise
- Column values are mapped back to the row-keys

Row Oriented
(RDBMS Model)

Multi-valued

id	Name	Age	Interests
1	Ricky		Soccer, Movies, Baseball
2	Ankur	20	
3	Sam	25	Music

null

Column Oriented
(Multi-value sorted map)

id	Name
1	Ricky
2	Ankur
3	Sam

id	Age
2	20
3	25

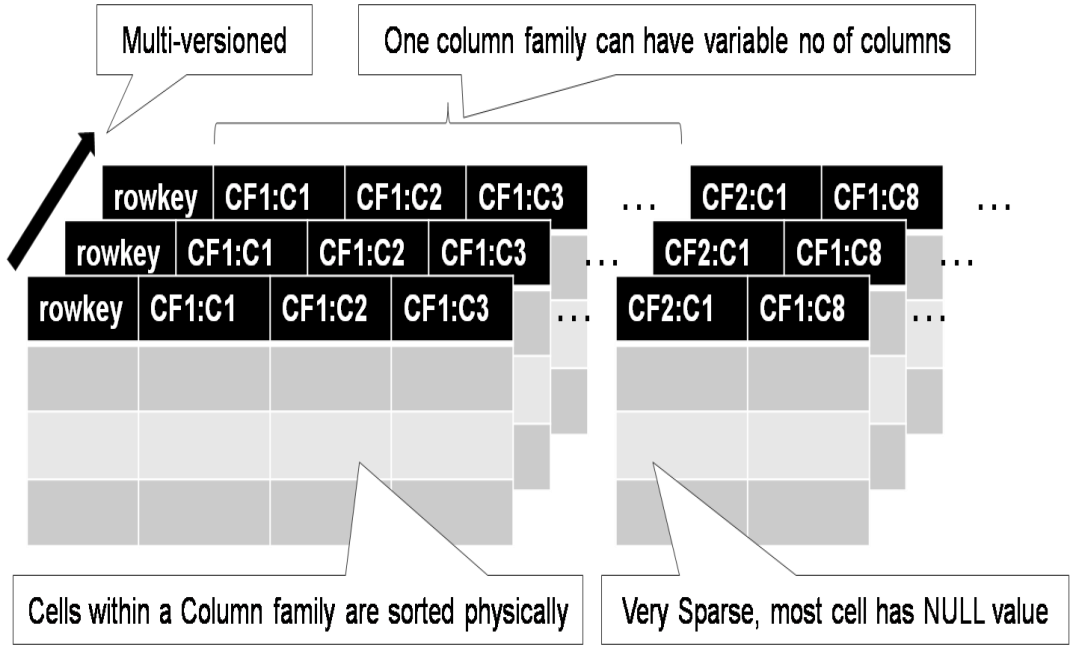
id	Interests
1	Soccer
1	Movies
1	Baseball
3	Music

Column Families

- Resembles a table in RDBMS
- Each column family can have more than one column
- Number of columns can vary for different rows

Row ID	Columns...		
1	**Name**	**Website**	
	Preston	www.example.com	
2	**Name**	**Website**	
	Julia	www.example.com	
3	**Name**	**Email**	**Website**
	Alice	example@example.com	www.example.com

Column Families



Column Family: User

rowid	Col_name	ts	Col_value
u1	name	v1	Ricky
u1	email	v1	ricky@gmail.com
u1	email	v2	ricky@ya
u2	name	v1	Sam
u2	phone	v1	650-3456

Column Family: Social

rowid	Col_name	ts	Col_value
u1	friend	v1	u10
u1	friend	v1	u13
u2	friend	v1	u10
u2	classmate	v1	u15

- One File per Column Family
- Data inside file is physically sorted
- Sparse: NULL cell does not materialize

Data Model in Cassandra

- Hybrid between key-value store and column oriented databases
- Column family - analog of a RDBMS table
- Row - identified uniquely by a key, has values as columns, all rows need not have same number of columns

Data Model in Cassandra (..contd)

- Keyspace - analog of a RDBMS schema, outermost container of data.
- Number of column families in Keyspace is fixed.
- Most basic attributes of a Keyspace are - Replication factor, Replica placement strategy

Data Distribution

- Rows are partitioned through a partition key which is the first component of a primary key
- Two ways to partition:
  - Random
  - Ordered

CQL (Cassandra Query Language)

- Way to interact with Cassandra
- Syntax is very similar to that of SQL, but far less capable
- No joins, no subqueries

Queries

- Create Keyspace - **CREATE KEYSPACE “users” WITH CREATE KEYSPACE “KeySpace Name” WITH replication = {'class': ‘Strategy name’, 'replication_factor' : ‘No.Of replicas’};**
- Consistency - **CONSISTENCY QUORUM**
- Capture - **CAPTURE ‘dest_file.txt’**
- Source - **SOURCE ‘myfile.txt’**
- Copy - **COPY airplanes (name, mach, year, manufacturer) TO 'temp.csv'**

Monitoring Cassandra Cluster

- Java Management Extension(JMX) can be used to monitor Cassandra cluster
- Several JMX compliant tools are available

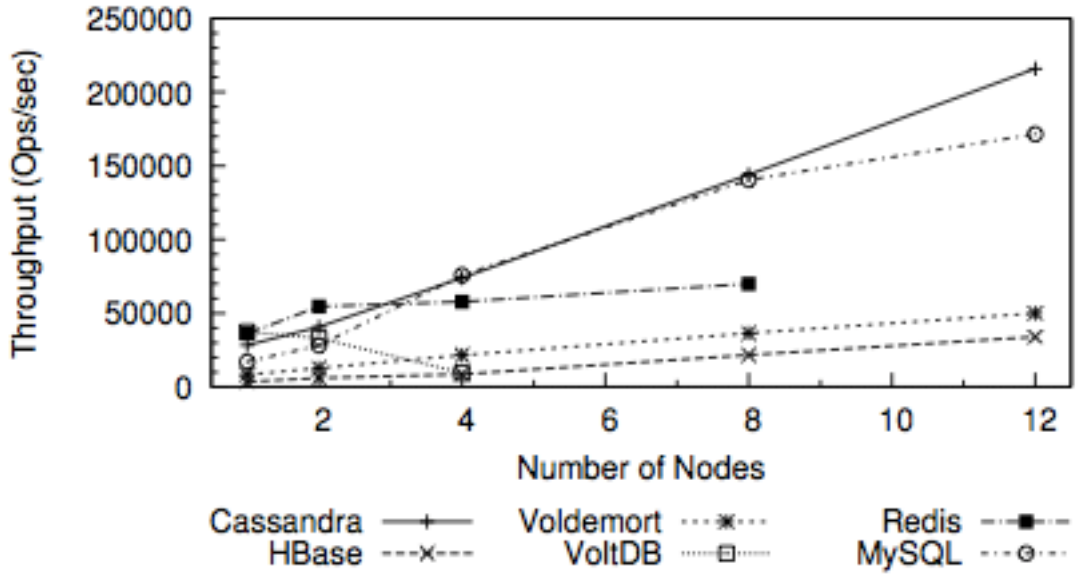
Applications

When to use Cassandra

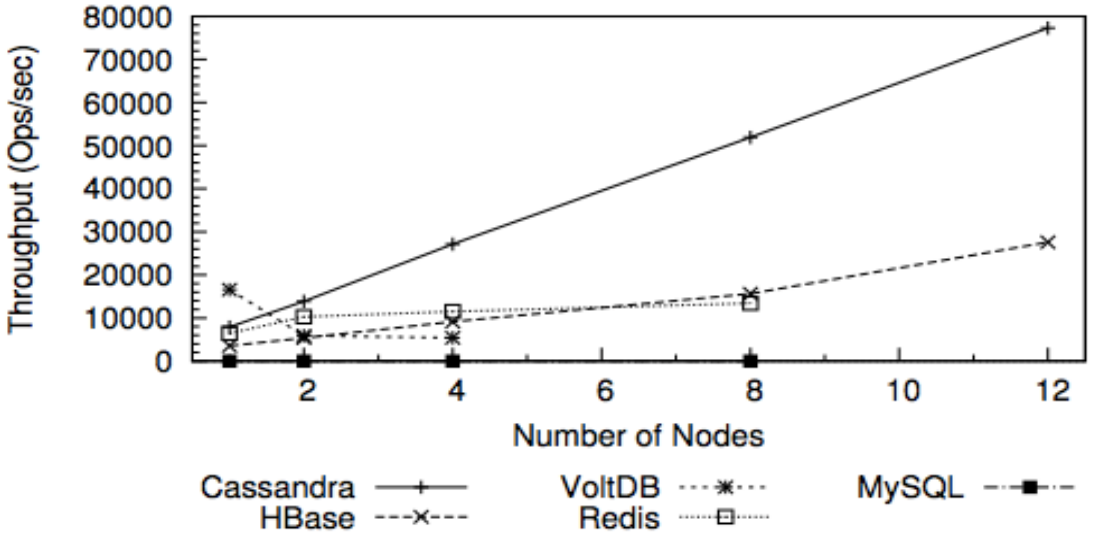
- Highly scalable.
- Reliable cross-datacenter replication
- Excellent choice for real-time Analytics workload. Faster write operations
- Higher insertion rates
- Can be integrated with Hadoop, Hive and Apache Spark for Batch Processing
- Tunable Consistency and CAP parameters.

Throughput comparison

Throughput for workload Read/Write

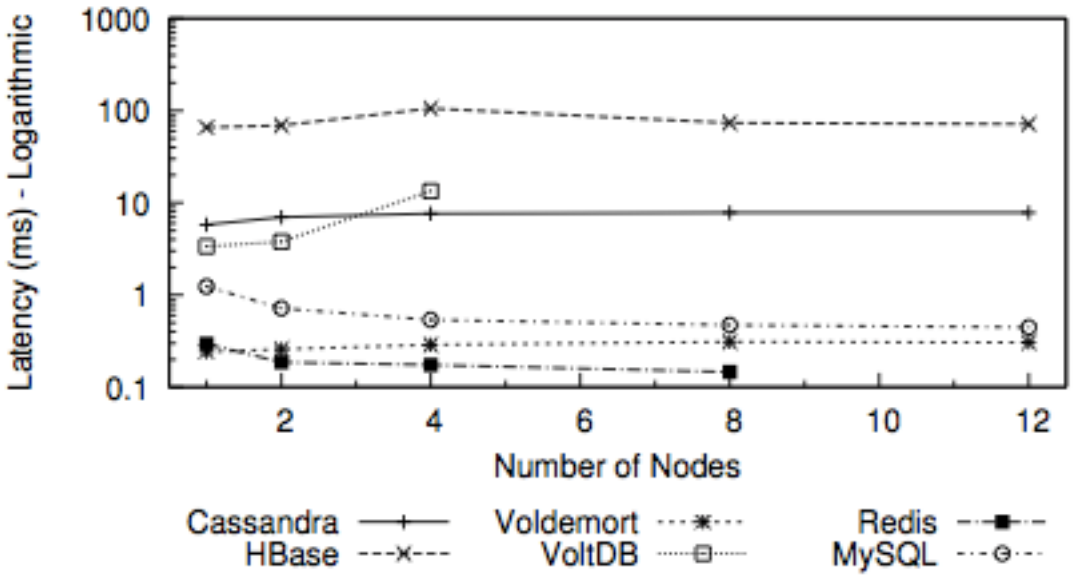


Throughput for workload Read/Scan/Write

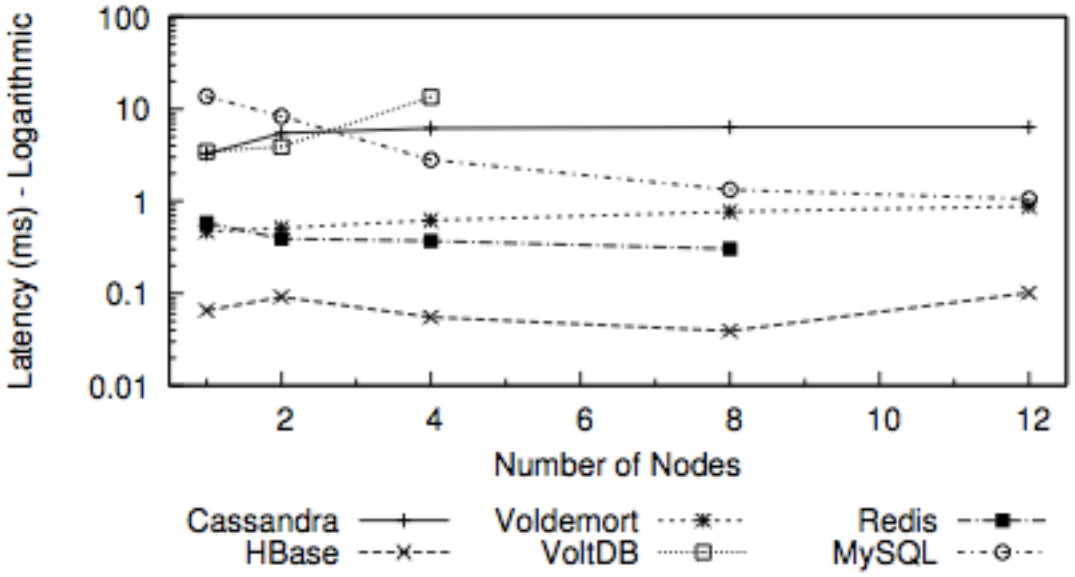


Tradeoffs

Read latency for workload Read/Write



Write latency for workload Read/Write



Drawbacks of Cassandra

- Transactions are not supported (ACID or otherwise)
- Eventual consistency isn’t sufficient always. Eg : Trading stocks.
- No support for ad-hoc queries
- Cannot perform complex queries

Connecting applications to Cassandra : Drivers

- Drivers help connect the applications to Cassandra database
- Driver languages :
  - Python
  - Java
  - C
  - C++
  - Ruby
  - And many more.
  - (Refer http://www.planetcassandra.org/apache-cassandra-client-drivers/)
- API’s are similar for all the languages

Setup

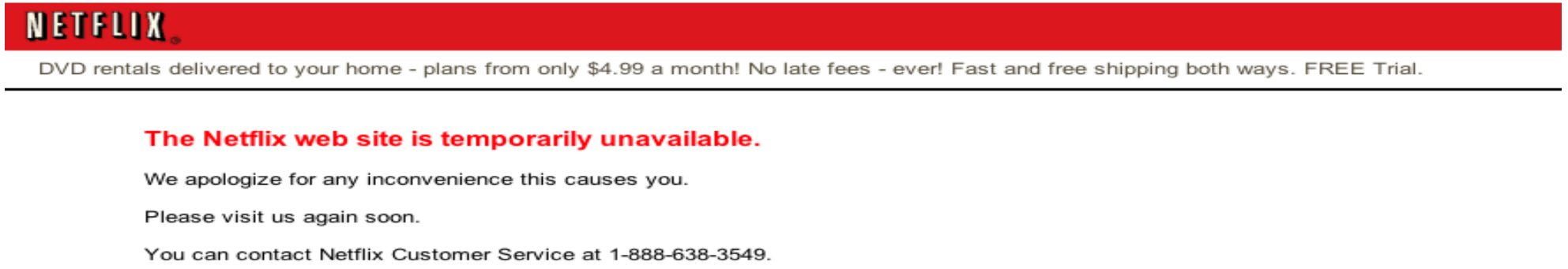
- Create a cluster object
- Use the cluster to obtain a session
- Session manages all the connections to the cluster

```
#connect to the cluster and the keyspace “sample”
from cassandra.cluster import Cluster
cluster= Cluster()
session=cluster.connect(‘sample’)
```

Session object listens to the changes in the cluster and the driver reacts to the same

NetFlix - Challenges faced :

- Single datacenter meant single point of failure
- Users grew exponentially
- And every 2 weeks , the site was down for maintenance



What was required ?

- More reliable and fault tolerant data storage
- High availability of member information, streaming quality video data in a more robust fashion
- Flexibility of streaming the video data from multiple devices

What Cassandra offered ?

- Created better business agility for Netflix.
- No downtime as there are no schemas.
- No fear of data loss because replication means no single point of failure .
- Open-source model provided Netflix the flexibility to implement their own backup, recovery system and replication strategy

Instagram : Shift from Redis to Cassandra

- Memory limitations!!
- Cut the costs to the point where they were paying around a quarter of what they were paying before.
- Primarily used for fraud detection, newsfeed and inbox

Facebook : Why Cassandra

- Operational Requirement : Performance, Reliability, Efficiency, High Scalability, Fault Tolerance.
- Cross datacenter replication
- Designed to address the storage needs of inbox search problem.
- Provided high write throughput
- Exploited the timestamp property provided by Cassandra

Facebook : Shift from Cassandra to HBase

- Eventual consistency model not suitable for the new messenger.
- Hbase – simpler consistency model
- High scalability and performance, auto load balancing.
- Hadoop – widely used by Facebook and HDFS being the distributed file system for both Hadoop and Hbase.